Scalable, Explainable and Provably Robust Anomaly Detection with One-Step Flow Matching

Zhong Li[♣,♡] **Qi Huang**[♣] **Yuxuan Zhu**[▲] **Lincen Yang**[♣(✉)]
Mohammad Mohammadi Amiri[▲] **Niki van Stein**[♣] **Matthijs van Leeuwen**[♣]

[♣]The Leiden Institute of Advanced Computer Science (LIACS), Leiden University
[▲]Department of Computer Science, Rensselaer Polytechnic Institute
[♡]The Intelligent Computing Research Center, Great Bay University
Corresponding Author (✉): l.yang@liacs.leidenuniv.nl (Lincen Yang)

Abstract

We introduce Time-Conditioned Contraction Matching (TCCM), a novel method for semi-supervised anomaly detection in tabular data. TCCM is inspired by flow matching, a recent generative modeling framework that learns velocity fields between probability distributions and has shown strong performance compared to diffusion models and generative adversarial networks. Instead of directly applying flow matching as originally formulated, TCCM builds on its core idea—learning velocity fields between distributions—but simplifies the framework by predicting a time-conditioned contraction vector toward a fixed target (the origin) at each sampled time step. This design offers three key advantages: (1) a lightweight and scalable training objective that removes the need for solving ordinary differential equations during training and inference; (2) an efficient scoring strategy called one time-step deviation, which quantifies deviation from expected contraction behavior in a single forward pass, addressing the inference bottleneck of existing continuous-time models such as DTE (a diffusion-based model with leading anomaly detection accuracy but heavy inference cost); and (3) explainability and provable robustness, as the learned velocity field operates directly in input space, making the anomaly score inherently feature-wise attributable; moreover, the score function is Lipschitz-continuous with respect to the input, providing theoretical guarantees under small perturbations. Extensive experiments on the ADBench benchmark show that TCCM strikes a favorable balance between detection accuracy and inference cost, outperforming state-of-the-art methods—especially on high-dimensional and large-scale datasets. The source code is provided at https://github.com/ZhongLIFR/TCCM-NIPS.

1 Introduction

Background. Anomaly detection in tabular data is the task of identifying data instances (or patterns) that deviate significantly from expected behavior (Chandola et al., 2009; Aggarwal and Aggarwal, 2017a; Pang et al., 2021). It has found widespread applications in various domains, such as fraud detection in finance (Hilal et al., 2022), fault detection in manufacturing (Yu and Zhang, 2023), intrusion detection in cybersecurity (Chou and Jiang, 2021), and medical diagnosis in healthcare (Fernando et al., 2021). In these high-stakes domains, data is growing rapidly in both size and dimensionality, calling for approaches that are not only effective but also scalable. Equally important, decisions made in these settings often have critical consequences, making interpretability an ethical and regulatory necessity (Li et al., 2023). Therefore, anomaly detection methods should also be able to provide meaningful explanations alongside accurate predictions.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

Positioning our work. Existing anomaly detection methods can be broadly categorized into classical machine learning approaches and deep learning-based techniques. Classical methods—such as OCSVM (Schölkopf et al., 1999), LOF (Breunig et al., 2000), PCA (Shyu et al., 2003), and KDE (Latecki et al., 2007)—often struggle with high-dimensional data due to the curse of dimensionality, and with large-scale datasets due to limited computational scalability. To address these limitations, deep learning-based anomaly detection methods have gained research attention and achieved strong performance across various domains (Pang et al., 2021). Deep methods can be grouped into two categories: (1) *two-stage approaches*, which first learn a low-dimensional representation (e.g., via an autoencoder) and then apply off-the-shelf anomaly detectors. However, such decoupled training strategies often struggle to learn task-effective features due to the lack of joint optimization (Nguyen and Vien, 2019); and (2) *end-to-end trained approaches*, which integrate representation learning and anomaly detection into a unified training objective, achieving better performance. Meanwhile, given that labeled anomalies are both scarce and expensive to obtain in many real-world applications—such as system failures, fraud, or clinical anomalies—many existing methods, both classical and deep, adopt a *semi-supervised* setting. In this paradigm, models are trained solely on normal data and tasked with identifying deviations at test time. Although claimed as"unsupervised" in some studies (Goodge et al., 2022), we rigorously refer to this setup as *semi-supervised anomaly detection* following An and Cho (2015); Ruff et al. (2018); Akcay et al. (2018); Bergman and Hoshen (2020). Our work is situated within this setting, adopting a deep[1], end-to-end, and semi-supervised approach that learns the structure of normality during training and identifies deviations at inference.

Limitations of Existing Studies. Despite recent advances in end-to-end deep anomaly detection, many existing approaches face fundamental limitations. Adversarial models such as AnoGAN (Schlegl et al., 2017) and GANomaly (Akcay et al., 2018) often suffer from training instability due to their reliance on min-max optimization. Density-based methods like DAGMM (Zong et al., 2018) introduce complex architectures to approximate latent distributions. Diffusion-based approaches such as Anomaly-DDPM and DTE (Livernoche et al., 2023) may depend heavily on carefully tuned noise schedules and sampling hyperparameters; in practice, they also suffer from extremely slow inference on large-scale datasets due to their iterative nature. Normalizing flows (e.g., OneFlow (Maziarka et al., 2021)) require invertibility and Jacobian computations, leading to trade-offs between model expressivity and computational efficiency. LUNAR (Goodge et al., 2022), which employs graph neural networks to capture relational structures, incurs high training costs and scales poorly with data size. Methods such as DeepSVDD (Ruff et al., 2018), which focus on compact representation learning, rely on restrictive architectural constraints (e.g., no biases, bounded activations) to avoid representation collapse, and often depend on numerous training heuristics to yield satisfactory results. Another major limitation lies in the lack of interpretability—most deep models offer little insight into why a sample is considered anomalous. While a few methods have made progress in this direction—e.g., AE-1SVM (Nguyen and Vien, 2019) using gradient-based attribution, ICL (Shenkar and Wolf, 2022) identifying key contributing features, MCM (Yin et al., 2024) modeling both feature-level abnormality and inter-feature correlations, and DTE (Livernoche et al., 2023) providing denoised reconstructions as explanations—such interpretable designs remain the exception rather than the norm. The vast majority of deep anomaly detection models continue to operate as black boxes, limiting their utility in high-stakes domains where interpretability is critical.

Flow Matching. Flow matching has emerged as a promising generative modeling framework that retains the training stability and expressivity of diffusion models, while offering improved computational efficiency (Liu et al., 2022; Lee et al., 2023). Instead of relying on stochastic differential equations (SDEs), it learns an ordinary differential equation (ODE) that deterministically maps samples from a source to a target distribution, enabling faster sampling and easier optimization. Flow matching also bypasses the need for a forward noising process and explicit density functions, making it well-suited for settings with implicit or intractable data distributions (Albergo and Vanden-Eijnden, 2023; CSAIL, 2024). Its interpolant formulation further allows empirical analysis of learned velocity fields over time (Albergo and Vanden-Eijnden, 2023), enhancing interpretability for downstream tasks. Despite its recent success in generative modeling, flow matching has not been explored for anomaly detection as of this writing, to the best of our knowledge.

[1]Following common practice in the machine learning community, we use the term "deep" to indicate deep learning-based, end-to-end models, even when the employed neural architecture is relatively shallow (e.g., a multi-layer perceptron with two hidden layers).

Contributions. Motivated by the limitations of existing deep anomaly detection methods and benefiting from recent advances in generative modeling—particularly flow matching—we introduce *Time-Conditioned Contraction Matching* (TCCM), a novel flow matching-inspired approach for semi-supervised anomaly detection. Specifically, TCCM learns a time-conditioned velocity field that contracts normal data, drawn from a source distribution ρ_{source}, towards a degenerate target distribution ρ_{target}, defined as a Dirac delta at the origin. Unlike previous approaches that simulate full continuous trajectories via ODE or SDE integration, TCCM avoids trajectory simulation entirely by directly learning a velocity field that approximates contraction dynamics from any input point at any time (details are described in Section 3). At test time, samples are scored based on how much their predicted velocity field deviates from the expected contraction pattern—an idea illustrated in Figure 1. This mismatch in velocity magnitudes and directions forms the basis of our anomaly score. TCCM inherits the scalability and simplicity of flow matching (Liu et al., 2022), and is trained using an unconstrained least-squares objective. It avoids adversarial instability (as in AnoGAN (Schlegl et al., 2017), GANomaly (Akcay et al., 2018)), complex density modeling (as in DAGMM (Zong et al., 2018) or KDE (Latecki et al., 2007)), and slow sampling-based inference (as in DTE (Livernoche et al., 2023)). Unlike normalizing flows (Maziarka et al., 2021), it requires neither invertibility nor Jacobian computation, and unlike DeepSVDD (Ruff et al., 2018), it does not rely on restrictive architectural constraints to avoid collapse. Furthermore, compared to graph-based methods like LUNAR (Goodge et al., 2022), TCCM achieves significantly faster training on large-scale datasets. Crucially, TCCM is inherently interpretable—its velocity field lives in the input space, supporting feature-wise attribution—and provably robust, with a Lipschitz-continuous anomaly score under small input perturbations.

Findings. We evaluate TCCM on 47 benchmark datasets from the ADBench suite (Han et al., 2022), comparing it against 44 baseline methods (23 deep learning-based and 21 classical), for a total of **10,575 runs** across five seeds. Our results demonstrate five key strengths: (1) *Accuracy*: TCCM achieves **top-1 performance** in both AUPRC and AUROC scores (see Appendix B.4 for definitions) across all evaluated methods (see Figures 2a and 2b for aggregated results). (2) *Scalability*: The model is highly efficient in both training and inference on high-dimensional and large-scale datasets—achieving, on average, **1573× faster inference** than DTE-NonParametric (top-2 in AUROC and AUPRC), and **85× faster inference** than LUNAR (top-3 in both metrics), while maintaining superior detection performance (see Figure 3a). (3) *Explainability*: TCCM enables feature-level attribution for anomaly scores, supporting interpretable diagnosis—an aspect largely absent in existing deep anomaly detection models (see Figure 4). (4) *Robustness*: We theoretically prove that the anomaly score satisfies a Lipschitz continuity condition, offering provable robustness guarantees under input perturbations (see Proposition 1). (5) *Simplicity of training*: TCCM requires no adversarial losses, density estimation, or noise schedules—making it simple to train, stable to optimize, and easy to reproduce (see Eq. 4). Together, these findings establish TCCM as a principled, highly effective, scalable, explainable, and provably robust solution for semi-supervised anomaly detection in tabular data.

2 Preliminaries

Due to space constraints, we defer a detailed discussion of related work—including anomaly detection methods and flow matching—to Appendix A, and begin with a general problem statement.

2.1 Problem Statement

Notations. Bold lowercase letters (e.g., $\boldsymbol{x}$) denote vectors; bold uppercase letters (e.g., $\boldsymbol{X}$) represent matrices. Calligraphic symbols (e.g., $\mathcal{X}$) denote sets, and standard italic letters (e.g., x) are used for scalars, unless otherwise specified. Besides, these symbols may be used to denote both random variables and their realizations; this dual use is common in the machine learning literature and will be made explicit whenever necessary.

Problem Setting. We consider a semi-supervised anomaly detection scenario, where only normal samples are available during training. Let $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^{N} \subset \mathbb{R}^d$ be a dataset of d-dimensional observations, partitioned into a training set $\mathcal{X}_{\text{train}}$ containing only normal instances sampled from an unknown distribution $p_{\text{data}}(\boldsymbol{x})$, and a test set $\mathcal{X}_{\text{test}}$ that may include both normal and anomalous samples.

Problem 1 (Semi-Supervised Anomaly Detection)**.** *Given access to normal training data $\mathcal{X}_{train} \subset \mathbb{R}^d$, the goal is to learn an anomaly scoring function $S : \mathbb{R}^d \to \mathbb{R}$ that quantifies the deviation of any test input $\boldsymbol{x} \in \mathcal{X}_{test}$ from the learned notion of normality.*

To solve this problem, we aim to learn the structure of normal data using only unlabeled normal instances. At test time, deviations from this learned structure are quantified and assigned anomaly scores, allowing the detection of abnormal inputs without access to anomalous data during training.

2.2 Recap of Flow Matching

Flow matching (or stochastic interpolant) (Lipman et al., 2022; Albergo and Vanden-Eijnden, 2023; Liu et al., 2022) provides a principled and flexible framework for learning neural ODE-based transport maps between two empirical distributions. Given samples from a source distribution $\boldsymbol{x}_0 \sim p_0$ and a target distribution $\boldsymbol{x}_1 \sim p_1$, the goal is to learn a time-dependent velocity field $v(\boldsymbol{x}_t, t)$ such that the following ODE governs the evolution between the two: $d\boldsymbol{x}_t = v(\boldsymbol{x}_t, t)dt$ for $t \in [0, 1]$. This Lagrangian formulation describes the motion of particles from $\boldsymbol{x}_0$ to $\boldsymbol{x}_1$, implicitly defining the coupling $\pi(p_0, p_1)$ between the distributions. The velocity field is parameterized as $v_\theta(\boldsymbol{x}_t, t)$ via a neural network and trained to match a reference velocity field using a simple least-squares objective:

$$\min_\theta \mathbb{E}_{t,\boldsymbol{x}_t} \left[\|v(\boldsymbol{x}_t, t) - v_\theta(\boldsymbol{x}_t, t)\|_2^2 \right]. \tag{1}$$

Different choices of interpolation path $\boldsymbol{x}_t$ and reference velocity $v(\boldsymbol{x}_t, t)$ give rise to different flow matching models. A widely used class of methods adopts the *probability flow ODE* formulation (Song et al., 2020), where the velocity incorporates the score function $\nabla \log p_t$ and corresponds to a deterministic trajectory derived from an underlying SDE. In this case, the path is often defined via a variance-preserving schedule:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sqrt{1 - \alpha_t^2}\, \boldsymbol{x}_1, \quad \text{with} \quad \alpha_t = \exp\left(-\frac{1}{2} \int_0^t \beta(s)\, ds \right),$$

where $\beta(s)$ is a pre-defined noise schedule that controls the rate of variance increase over time. This form allows equivalence to score matching under certain conditions (Lee et al., 2023; Zheng et al., 2023), and is popular in diffusion-based generative models. However, the resulting curved trajectory can complicate optimization and slow down sampling (Liu et al., 2022).

To address these issues, Liu et al. (2022) have proposed a *constant velocity ODE* approach, where the interpolation path is simply linear: $\boldsymbol{x}_t = (1 - t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$. In this case, the reference velocity becomes a constant vector $\boldsymbol{x}_1 - \boldsymbol{x}_0$, and the flow matching objective reduces to:

$$\min_\theta \mathbb{E}_{t,\boldsymbol{x}_t} \left[\|\boldsymbol{x}_1 - \boldsymbol{x}_0 - v_\theta(\boldsymbol{x}_t, t)\|_2^2 \right]. \tag{2}$$

This variant is known as the *rectified flow* model, and has been shown to improve training efficiency and reduce curvature in the learned trajectories, facilitating both forward simulation and backward sampling. Our proposed method builds on this formulation, leveraging its simplicity and scalability while adapting it for the anomaly detection setting.

3 Methodology: Time-Conditioned Contraction Matching (TCCM)

Core idea. Conventional flow matching models (Lipman et al., 2022; Liu et al., 2022) construct continuous-time trajectories that gradually transport samples from a source distribution (at $t = 0$) to a target distribution (at $t = 1$) by integrating a learned velocity field over the entire time interval. In contrast, our method departs from this paradigm both conceptually and technically. Rather than relying on the full trajectory across time to learn the transformation, we directly learn a *contraction vector field* at each time step—one that immediately points from the current position to the fixed target (the origin). This allows the model to predict the contraction behavior independently at each time point, avoiding the need for simulating or reconstructing the full flow path. This provides a powerful yet simple framework for anomaly detection: every point learns how to contract back to the origin over time, which motivates the name *Time-Conditioned Contraction Matching* (TCCM).

Formally, we treat the data distribution as the source, $\boldsymbol{z}_0 := \boldsymbol{z} \sim p_{\text{data}}$, and consider the target as a degenerate Dirac distribution at the origin, $\boldsymbol{z}_1 := \mathbf{0}$. While this setup may suggest a flow-like

interpretation, we emphasize that our model is not tasked with approximating the full solution of a dynamical system such as:

$$d\boldsymbol{z}(t) = -\boldsymbol{z}(t)dt, \quad \text{with} \quad \boldsymbol{z}(0) = \boldsymbol{z}, \tag{3}$$

whose analytical solution would be $\boldsymbol{z}(t) = \boldsymbol{z} \cdot e^{-t}$. However, our model does not supervise or simulate $\boldsymbol{z}(t)$ across time. Instead, we adopt a simplified training strategy that uses a constant target direction $-\boldsymbol{z}$ for supervision at all time steps. To achieve this, we learn a neural velocity field $f_{\boldsymbol{\theta}}(\cdot)$ on an augmented space $\tilde{\boldsymbol{z}} = [\boldsymbol{z}; \text{Embed}(t)]$. Specifically, $f_{\boldsymbol{\theta}}(\cdot)$ is conditioned on both the input $\boldsymbol{z}$ and a time variable $t \in [0, 1]$. The time is encoded using sinusoidal embeddings (Vaswani et al., 2017), which are concatenated with the input: $\tilde{\boldsymbol{z}} = [\boldsymbol{z}; \text{Embed}(t)]$, and passed through the model $f_{\boldsymbol{\theta}}$ to predict a contraction vector.

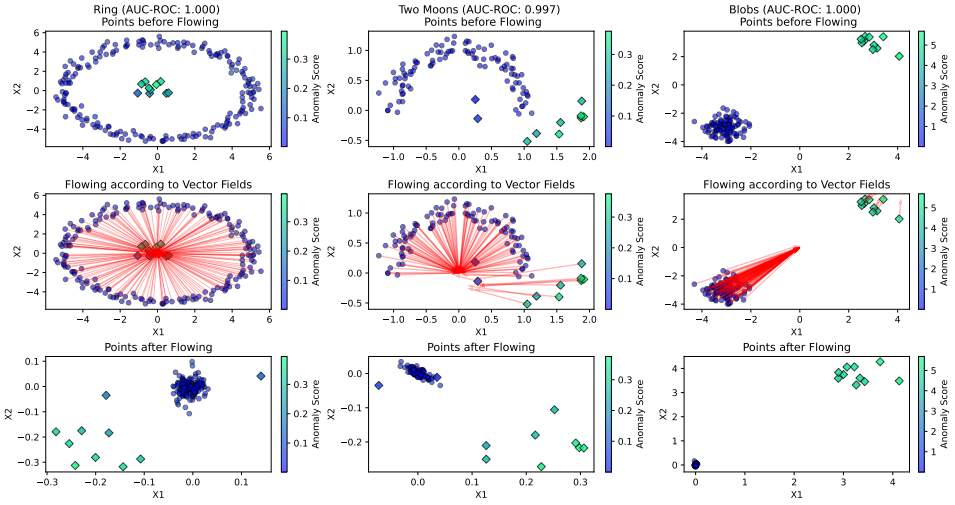


Figure 1: **Core idea of TCCM:** TCCM learns a time-conditioned velocity (vector) field that contracts normal data points, sampled from a source distribution ρ_{source}, towards a degenerate target distribution ρ_{target}, defined as a Dirac delta at the origin. At test time, anomalies are detected by measuring inconsistency with this learned contraction field. **Illustrative examples:** We visualize TCCM behavior on synthetic 2D datasets with varying normal (circles) and anomalous (squares) distributions. *Left*: Normal data form a ring; anomalies are sampled from a central Gaussian. *Middle*: Normals follow an upper moon; anomalies form a sparse lower moon. *Right*: Normals are clustered bottom-left; anomalies are drawn from a distinct Gaussian in the top-right. In all cases, TCCM successfully distinguishes anomalies based on their deviation from the expected contraction vector.

Training Objective. The training minimizes the following loss:

$$\min_{\theta} \mathbb{E}_{\boldsymbol{z} \sim p_{\text{data}},\, t \sim \mathcal{U}(0,1)} \left[\| f_{\boldsymbol{\theta}}([\boldsymbol{z}; \text{Embed}(t)]) + \boldsymbol{z} \|_2 \right]. \tag{4}$$

Optimizing objective (4) encourages the model to predict a velocity vector that approximates the negation of the current state, i.e., $f_{\boldsymbol{\theta}}([\boldsymbol{z}; \text{Embed}(t)]) \approx -\boldsymbol{z}$. This guides the system to evolve toward the origin by learning both the direction and magnitude of motion in a time-dependent manner. To achieve this, the neural velocity field is required to extract the common factors of variation present in normal data. As a result, normal samples following their predicted velocity fields can approach the origin at any given time, while anomalous instances, due to their deviation from the learned structure, fail to do so. The pseudocode for training is given in Algorithm 1 in Appendix B.5.

Interpretation and Motivation. Although not derived from an explicit ODE, our method can be viewed as learning a time-aware vector field that approximates the contraction dynamics toward a shared target. This formulation provides several advantages: *(1) Time-Conditioned Consistency:* The model learns to predict contraction vectors across time steps that consistently guide inputs toward the origin, promoting geometric alignment and stability; *(2) Simplified Supervision:* Using a fixed supervision target $-\boldsymbol{z}$ removes the need for trajectory supervision, leading to a simpler and smoother optimization process; *(3) No ODE Solvers Required:* Unlike conventional flow-based models, TCCM avoids numerical integration during both training and inference, resulting in substantial computational

efficiency; *(4) Learnable Temporal Dynamics:* Sinusoidal time embeddings allow the model to modulate both the magnitude and direction of contraction vectors over time, enabling rich, non-linear temporal behavior.

Anomaly Scoring at Inference Time. Given a test input $\boldsymbol{z} \in \mathcal{X}_{\text{test}}$ and a fixed evaluation time $t_{\text{fixed}} \in (0, 1]$, we define the anomaly score as:

$$S_{\text{fixed}}(\boldsymbol{z}; t_{\text{fixed}}) = \left\| f_{\boldsymbol{\theta}} \left([\boldsymbol{z}; \text{Embed}(t_{\text{fixed}})] \right) + \boldsymbol{z} \right\|_2 , \tag{5}$$

where $f_{\boldsymbol{\theta}}([\boldsymbol{z}; \text{Embed}(t)])$ denotes the learned velocity field conditioned on both the input feature $\boldsymbol{z}$ and the time variable t, encoded via sinusoidal embeddings and concatenated with $\boldsymbol{z}$ before being passed into a multilayer perceptron (MLP). This scoring strategy is grounded in the following expectations: (1) *Normal instances* are trained to follow a contraction path toward the origin. Since supervision is based on a constant target $-\boldsymbol{z}$ across all time steps, a well-aligned normal sample satisfies $f_{\boldsymbol{\theta}}([\boldsymbol{z}; \text{Embed}(t)]) \approx -\boldsymbol{z}$, leading to a small residual norm. (2) *Anomalous instances*, which deviate from the learned contraction pattern, yield misaligned velocities and hence higher residuals. (3) This approach is computationally efficient, as it avoids solving ODEs and requires only a single forward pass through the network at a chosen time step t_{fixed}, which overcomes the primary bottleneck of high evaluation cost found in existing continuous-time ODE/SDE models such as Anomaly-DDPM (Livernoche et al., 2023) and DTE-NonParametric (Livernoche et al., 2023). (4) Importantly, because the residual vector $f_{\boldsymbol{\theta}}([\boldsymbol{z}; \text{Embed}(t_{\text{fixed}})]) + \boldsymbol{z}$ lies in the original feature space, the absolute values of its entries directly quantify how much each feature contributes to the anomaly score. This provides intrinsic feature-level interpretability, in contrast to post-hoc explanation methods such as SHAP (Lundberg and Lee, 2017) and LIME (Ribeiro et al., 2016).

We refer to this anomaly scoring procedure as *one-step flow matching* because, unlike classical flow matching models that integrate velocity fields across time to compute transformation paths, our method makes a single-time-point evaluation to determine alignment with the learned contraction dynamics. While the underlying model is termed *TCCM*, this scoring mechanism captures the spirit of flow matching—comparing learned dynamics to an ideal contraction vector—yet does so in a highly scalable one-step formulation. Although the evaluation time t_{fixed} in Eq 5 can be any value in $(0, 1]$, we set $t_{\text{fixed}} = 1$ by default throughout our experiments for simplicity. Particularly, we provide a sensitivity analysis (see Figure 13) showing that the anomaly detection performance is largely stable across different values of t, validating the temporal consistency of the learned flow field and its ability to produce meaningful predictions at any time step. The pseudocode for inference is given in Algorithm 2 in Appendix B.5.

4 Theoretical Properties of TCCM

In this section, we establish two key theoretical properties of our method: *(i)* Lipschitz continuity of the anomaly score, which leads to provable robustness guarantees under input perturbations; and *(ii)* discriminative behavior of the score function under distributional shift, explained via a stylized Gaussian mixture setting. These results offer both certifiability of robustness and theoretical insight into the score function's discriminative behavior, complementing our empirical findings.

Proposition 1 (Lipschitz Continuity and Robustness)**.** *Let $f_{\boldsymbol{\theta}}(\cdot, t_{fixed})$ be L-Lipschitz continuous in its first argument (for a fixed time $t_{fixed} \in (0, 1]$). Then the anomaly score*

$$S_{fixed}(\boldsymbol{x}) := \left\| f_{\boldsymbol{\theta}} \left([\boldsymbol{x}; Embed(t_{fixed})] \right) + \boldsymbol{x} \right\|_2$$

is $(L + 1)$-Lipschitz continuous with respect to $\boldsymbol{x}$, i.e.,

$$|S_{fixed}(\boldsymbol{x}_1) - S_{fixed}(\boldsymbol{x}_2)| \leq (L + 1)\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2.$$

Proof. Define $g(\boldsymbol{x}) := f_{\boldsymbol{\theta}}([\boldsymbol{x}; \text{Embed}(t_{\text{fixed}})]) + \boldsymbol{x}$. Then:

$$\|g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2)\|_2 \leq \|f_{\boldsymbol{\theta}}([\boldsymbol{x}_1; \text{Embed}(t)]) - f_{\boldsymbol{\theta}}([\boldsymbol{x}_2; \text{Embed}(t)])\|_2 + \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2 \leq (L+1)\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2.$$

Finally, since the ℓ_2 norm is 1-Lipschitz, we have $|S_{\text{fixed}}(\boldsymbol{x}_1) - S_{\text{fixed}}(\boldsymbol{x}_2)| \leq \|g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2)\|_2$. □

Remark and Implications. The assumption that $f_{\boldsymbol{\theta}}(\cdot, t_{\text{fixed}})$ is Lipschitz is both theoretically and practically reasonable. In continuous normalizing flows (CNFs) and flow-matching models, such smoothness is often required to ensure existence and uniqueness of solutions (via Picard–Lindelöf

theorem (Murray and Miller, 2013)) or to stabilize ODE solvers. More specifically, the function $f_{\boldsymbol{\theta}}$ is implemented as a multilayer perceptron with ReLU activations and a fixed architecture, making it piecewise linear and hence Lipschitz continuous. The Lipschitz constant L can be further controlled through spectral normalization, gradient penalties, or other regularization techniques. Moreover, this proposition has the following two **implications**: (1) *robustness:* the score is stable under small perturbations, enhancing reliability in noisy or adversarial environments; and (2) *certifiability:* the bound implies that $|S(\boldsymbol{x}+\delta) - S(\boldsymbol{x})| \leq (L+1)\varepsilon$ if $\|\delta\| \leq \varepsilon$, providing a certifiable safety margin.

To theoretically support the discriminative power of our anomaly score, we analyze an idealized setting where normal and anomalous instances are drawn from two disjoint Gaussian mixture models (GMMs) with shared isotropic covariance. Although simplified, this setup enables a clean analysis of how the learned score function behaves on out-of-distribution samples. Under the assumption that the model has learned a noisy contraction field of the form $f_{\boldsymbol{\theta}}([\boldsymbol{x}; \text{Embed}(1)]) = -\boldsymbol{x} + \boldsymbol{\epsilon}$ for normal training data, we establish the following result:

Proposition 2 (Discriminative Power under GMM-to-GMM Shift)**.** *Let normal samples be drawn from a Gaussian mixture $p_{normal}(\boldsymbol{x}) = \sum_{r=1}^{R} \pi_r \cdot \mathcal{N}(\boldsymbol{\mu}_r, \sigma^2 I_d)$, and anomalous samples from a disjoint mixture $p_{anom}(\boldsymbol{z}) = \sum_{s=1}^{S} \eta_s \cdot \mathcal{N}(\boldsymbol{\nu}_s, \sigma^2 I_d)$, with $\boldsymbol{\nu}_s \notin \{\boldsymbol{\mu}_r\}_{r=1}^{R}$. Assume the learned contraction field satisfies $f_{\boldsymbol{\theta}}([\boldsymbol{x}; Embed(1)]) = -\boldsymbol{x} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d)$; and the learned velocity field is mismatched for anomalies. Define the anomaly score: $S(\boldsymbol{x}) = \|f_{\boldsymbol{\theta}}([\boldsymbol{x}; Embed(1)]) + \boldsymbol{x}\|_2 = \|\boldsymbol{\epsilon}\|_2$. Then, it holds that: (1) for normal samples, $S(\boldsymbol{x}) \sim \chi_d \cdot \sigma_f$, and $\mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}$; (2) for anomalies, let $\lambda_s = \frac{\|\boldsymbol{\nu}_s - \boldsymbol{\mu}_{r^*(s)}\|_2^2}{\sigma_f^2}$, where $r^*(s) := \arg\min_r \|\boldsymbol{\nu}_s - \boldsymbol{\mu}_r\|_2$, we have $S(\boldsymbol{z}) \sim \sum_{s=1}^{S} \eta_s \cdot \chi_d(\lambda_s)$; and (3) the expected anomaly scores of normal and anomalous instances satisfy: $\mathbb{E}[S(\boldsymbol{z})] > \mathbb{E}[S(\boldsymbol{x})]$. This implies that our score function assigns, in expectation, higher values to anomalies than to normal points—providing a theoretical foundation for its discriminative capability.*

The proof, provided in Appendix C.2, shows that the anomaly score corresponds to the norm of a central chi-distributed variable for normal samples and a non-central chi-distributed one for anomalies. The non-centrality parameter captures the squared distance between each anomaly and the closest normal-mode center, leading to systematically larger scores.

Implications. This result provides a theoretical lens into why our anomaly score increases for distributional outliers. Even though real-world data may not exactly follow Gaussian mixtures, the underlying intuition persists: samples that deviate from the structure captured by the contraction field are naturally assigned larger residuals.

In addition, Appendix C.3 further analyzes the model's representation dynamics and verifies that TCCM avoids degenerate or collapsed mappings in practice, complementing the above theoretical guarantees with empirical evidence of stable and discriminative behavior.

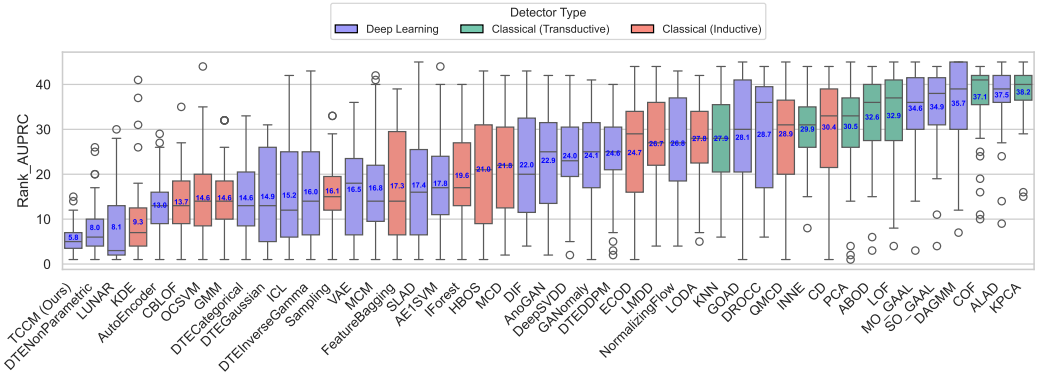
5 Experiments

We conduct comprehensive experiments to address the following research questions: (1) **Effectiveness**—Can TCCM outperform existing baselines in anomaly detection? (2) **Scalability**—How does TCCM compare to the strongest baselines in detection accuracy in terms of training and inference efficiency? (3) **Explainability**—Are the explanations generated by TCCM intuitive and meaningful to human users? (4) **Ablation Studies and Sensitivity Analysis**—How do various design choices impact the performance of TCCM?

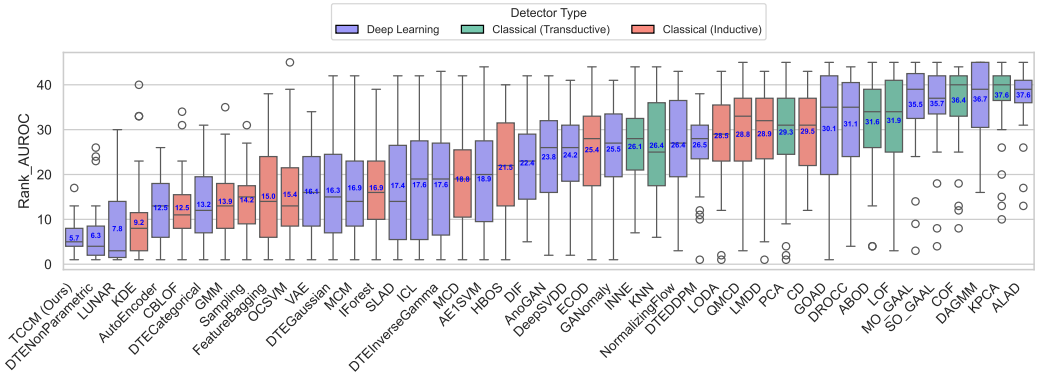
5.1 Experiment Setup

Datasets Description and Processing. (1) **Dataset Description**: A summary of the datasets used in our study is provided in Table 1. We adopt 47 benchmark datasets from the well-established ADBENCH benchmark (Han et al., 2022), spanning diverse domains including sociology, finance, linguistics, physics, and healthcare. To enable a comprehensive evaluation of different anomaly detectors, including our proposed method, we categorize the datasets into four groups based on their scale and dimensionality: (a) *High-dimensional* datasets, with more than 50 features; (b)

Large-scale datasets (but not high-dimensional), containing more than 10,000 instances and fewer than 50 features; (c) *Medium-scale* datasets (not high-dimensional), with 1,000 to 10,000 instances; and (d) *Small-scale* datasets (not high-dimensional), containing fewer than 1,000 instances. This categorization facilitates a nuanced analysis of model performance across varying data regimes. (2) **Data Processing**: We adopt a semi-supervised anomaly detection setting, where models are trained solely on normal instances. Specifically, we apply a stratified split to the normal data, using 50% for training and holding out the rest for testing. The test set includes both normal and anomalous samples. All features are standardized using a `StandardScaler` (Pedregosa et al., 2011) fitted on the training data (see Figure 16 in Appendix D.3 for an ablation study on the effect of feature normalization). This protocol is consistent with common practices in anomaly detection (e.g., (Zong et al., 2018; Bergman and Hoshen, 2020; Shenkar and Wolf, 2022; Yin et al., 2024)) and ensures a fair evaluation.



(a) AUPRC ranking distribution across 47 datasets for 45 anomaly detectors.



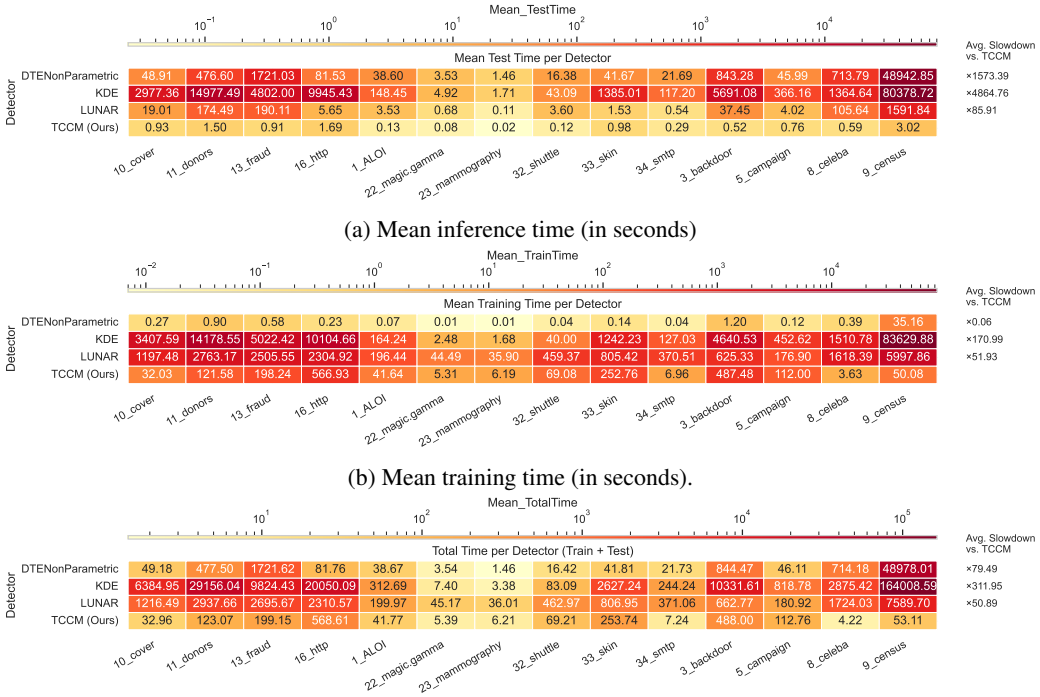
(b) AUROC ranking distribution across 47 datasets for 45 anomaly detectors..

Figure 2: Box plots of detector rankings based on AUPRC and AUROC scores across 47 datasets. Medians are marked by horizontal lines; means are shown as numbers.

Baselines and Evaluation Metrics. (1) **Baselines**: We evaluate our method against 44 baselines, including 21 classical (shallow) and 23 deep anomaly detection algorithms. Detailed descriptions of these baselines are provided in Appendix B.2. In particular, we offer a critical review of each deep method, highlighting their limitations in comparison to our approach in Appendix A.1. (2) **Evaluation metrics**: We adopt two standard metrics—Area Under the Receiver Operating Characteristic curve (AUROC) and Area Under the Precision-Recall Curve (AUPRC)—with higher values indicating better performance (see Appendix B.4 for more information).

Configurations. The details of architectures and hyperparameters will be postponed to Appendix B.3, while we highlight some of the main characteristics of our model here: the vector field $f_\theta(\boldsymbol{x}, t)$ is parameterized by a 3-layer multilayer perceptron (MLP), where each hidden layer contains 256 units followed by ReLU activations. To incorporate time information, we use a fixed sinusoidal embedding of the scalar time input $t \in [0, 1]$, following the positional encoding scheme used in transformer models (Vaswani et al., 2017). The time embedding is concatenated with the input vector $\boldsymbol{x}$, and the combined representation is passed through the MLP to produce the predicted velocity field.

5.2 Results Analysis



(a) Mean inference time (in seconds)

(b) Mean training time (in seconds).

(c) Mean total time (in seconds).

Figure 3: Mean run time (in seconds) across large-scale datasets for TCCM and other top-performing baselines in detection accuracy.

(1) Effectiveness. Figures 2a and 2b present the aggregated results based on AUPRC and AUROC scores, respectively. Due to the large scale of our experiments—covering 45 anomaly detectors across 47 datasets with 5 different random seeds, resulting in a total of 10,575 runs—it is impractical to include all individual results in the main paper. We thus report the complete results in Tables 6–13 in Appendix D. Particularly, we evaluate each method by reporting the distribution of its rankings across the 47 datasets. Rankings are computed based on the average AUPRC (respectively, AUROC) across the 5 seeds. As shown in Figures 2a and 2b, our method, TCCM, achieves the best overall performance in terms of both AUPRC (with an average rank of 5.8) and AUROC (with an average rank of 5.7). While DTE-NonParametric (second in both AUPRC and AUROC), LUNAR (third in both), and KDE (fourth in both) also demonstrate strong detection accuracy, we will show later that these methods suffer from poor scalability in training and/or inference, making them less favorable for large-scale deployment compared to TCCM. A more detailed analysis is deferred to Appendix D.1 due to space constraint. We further perform statistical significance testing using the Friedman (Friedman, 1937) and Nemenyi tests (Nemenyi, 1963) to assess whether the observed ranking differences are statistically meaningful; detailed results are provided in Appendix D.5.

(2) Scalability. TCCM achieves considerably faster inference than most deep learning baselines, particularly on large-scale and high-dimensional datasets. As shown in Figure 3a, it significantly outpaces other *high-accuracy* methods in inference speed—being 1,573.39× faster than DTE-NonParametric, 4,864.76× faster than KDE, and 85.91× faster than LUNAR on average. On the largest dataset, *census* (299,285 samples × 500 dimensions), TCCM takes just 1.50 seconds, while DTE-NonParametric requires 48,942 seconds. These results highlight TCCM's scalability and suitability for real-time anomaly detection in big-data environments. Beyond inference efficiency, we further provide an analysis on training time and total runtime to evaluate the end-to-end deployability of TCCM. As shown in Figure 3b, TCCM maintains competitive training efficiency—while DTE-NonParametric trains faster (requiring only 0.06× the training time of TCCM), KDE and LUNAR are **170.99**× and **51.93**× slower, respectively. When considering the overall cost, TCCM exhibits the lowest total runtime among all top-performing baselines (Figure 3c), outperforming DTE-NonParametric, KDE,

and LUNAR by **79.49**×, **311.95**×, and **50.89**×, respectively. This balanced efficiency across both training and inference phases underscores TCCM's suitability for real-world, large-scale anomaly detection deployments, where both accuracy and runtime constraints are critical. To further contextualize the trade-off between speed and performance, we include scatter plots comparing average inference time versus average AUROC (or AUPRC) across all 44 baselines (see Figures 7 and 8 in Appendix D.2.1). The results demonstrate that TCCM achieves one of the best balances between detection accuracy and inference efficiency among all evaluated methods. A detailed breakdown and additional comparisons across all 45 anomaly detection methods are provided in Appendix D.2.

(3) Explainability. TCCM is designed for tabular data and inherently supports self-explanation by producing feature-wise importance scores derived from its learned residual velocity field, which characterizes deviation from expected normal contraction behavior. To provide a more intuitive illustration of this property, we apply TCCM to image data (MNIST (Deng, 2012)), treating each pixel as a feature in a flattened tabular vector. We use digit `1` as the normal class and digit `7` as the anomaly (achieving an AUROC of 0.76). As shown in Figure 4, the model highlights the additional horizontal stroke that distinguishes `7` from `1`, demonstrating that the learned importance scores align well with human-interpretable cues. Importantly, these explanations are *intrinsic* to TCCM rather than post hoc approximations such as SHAP (Lundberg and Lee, 2017) or LIME (Ribeiro et al., 2016): the residual vector itself encodes per-feature contributions to the anomaly score, faithfully reflecting the model's internal reasoning. For this, we provide a controlled synthetic study in Appendix D.4.2, which quantitatively validates the accuracy of these feature-level attributions and further substantiates TCCM's intrinsic interpretability. This makes the explanations directly actionable in practice, enabling domain experts in areas such as fraud detection, healthcare, or industrial monitoring to identify not only *which* instances are anomalous but also *why*.

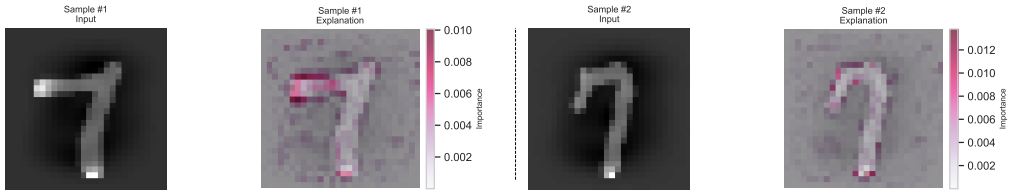


Figure 4: Illustrative examples of anomalous images and their explanations, where digit '1' is treated as the normal class and digit '7' as the anomaly. The highlighted regions correspond to structural differences between '7' and '1', which the model identifies as key contributors to the anomaly score.

(4) Ablation Studies and Sensitivity Analysis. We study how major design and data factors influence TCCM (see Appendix D.3): (1) Time embedding and inference time (Figures 12–13): results are nearly unchanged across choices, showing strong robustness; (2) Noise injection (Figure 14): deterministic training consistently performs better; (3) Training contamination (Figure 15): higher anomaly ratios reduce accuracy, underscoring the need for clean supervision; (4) Feature normalization (Figure 16): z-score normalization is generally beneficial and improves robustness; (5) Time-interpolated inputs (Figure 17): interpolation offers no gain and may add noise; (6) Comparison with Autoencoder +Time Embedding: confirms that TCCM learns a time-conditioned velocity field rather than a reconstruction mapping. Overall, TCCM remains stable and efficient across all variations.

6 Conclusion

We presented *Time-Conditioned Contraction Matching* (TCCM), a novel method for semi-supervised anomaly detection in tabular data. By learning a time-conditioned contraction field grounded in flow matching, TCCM avoids adversarial training, trajectory simulation, and density modeling—offering a lightweight yet expressive alternative to existing generative-based anomaly detection methods. On the ADBench benchmark, TCCM outperforms 44 classical and deep baselines in both AUROC and AUPRC, while achieving orders-of-magnitude faster inference than the strongest diffusion-based competitor (namely DTE-NonParametric). It also provides feature-level interpretability via its learned vector field. Theoretical analysis confirms the Lipschitz continuity and discriminative power of its scoring function. Together, these results position TCCM as an highly effective, scalable, interpretable, and robust solution for large-scale anomaly detection. Future directions include extending to other data modalities. The limitations and broader impacts of our work are further discussed in Appendix D.6.

Acknowledgment

We thank all anonymous reviewers for their time and efforts in reviewing this paper and their constructive comments to improve this paper. **Qi Huang, Niki van Stein**: This publication is partly sponsored by the XAIPre project (with project number 19455) of the research program Smart Industry 2020 which is (partly) financed by the Dutch Research Council (NWO).

References

Deepak Agarwal. 2007. Detecting anomalies in cross-classified streams: a bayesian approach. *Knowledge and information systems* 11, 1 (2007), 29–44.

Charu C Aggarwal and Charu C Aggarwal. 2017a. *An introduction to outlier analysis*. Springer.

Charu C Aggarwal and Charu C Aggarwal. 2017b. *Outlier ensembles*. Springer.

Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. 2018. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian conference on computer vision*. Springer, 622–637.

Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery* 29, 3 (2015), 626–688.

Michael S Albergo and Eric Vanden-Eijnden. 2023. Building Normalizing Flows with Stochastic Interpolants. In *11th International Conference on Learning Representations, ICLR 2023*.

Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE* 2, 1 (2015), 1–18.

Fabrizio Angiulli and Clara Pizzuti. 2002. Fast outlier detection in high dimensional spaces. In *European conference on principles of data mining and knowledge discovery*. Springer, 15–27.

Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. 1996. A Linear Method for Deviation Detection in Large Databases.. In *KDD*, Vol. 1141. 972–981.

Tharindu R Bandaragoda, Kai Ming Ting, David Albrecht, Fei Tony Liu, Ye Zhu, and Jonathan R Wells. 2018. Isolation-based anomaly detection using nearest-neighbor ensembles. *Computational Intelligence* 34, 4 (2018), 968–998.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

Liron Bergman and Yedid Hoshen. 2020. Classification-based anomaly detection for general data. *arXiv preprint arXiv:2005.02359* (2020).

Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano. 2021. A review on outlier/anomaly detection in time series data. *ACM computing surveys (CSUR)* 54, 3 (2021), 1–33.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 93–104.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 1–58.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. *Advances in neural information processing systems* 31 (2018).

Xin Chen and Anderson Ye Zhang. 2024. Achieving optimal clustering in Gaussian mixture models with anisotropic covariance structures. *Advances in Neural Information Processing Systems* 37 (2024), 113698–113741.

Dylan Chou and Meng Jiang. 2021. A survey on data-driven network intrusion detection. *ACM Computing Surveys (CSUR)* 54, 9 (2021), 1–36.

R Dennis Cook. 1977. Detection of influential observation in linear regression. *Technometrics* 19, 1 (1977), 15–18.

MIT CSAIL. 2024. Introduction to Flow Matching and Diffusion Models. MIT Computer Science Class 6.S184: Generative AI with Stochastic Differential Equations. `https://diffusion.csail.mit.edu/` Accessed: March 9, 2025.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. 2023. Flow matching in latent space. *arXiv preprint arXiv:2307.08698* (2023).

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* 7, Jan (2006), 1–30.

Li Deng. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine* 29, 6 (2012), 141–142.

Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516* (2014).

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. 2022. Genie: Higher-order denoising diffusion solvers. *Advances in Neural Information Processing Systems* 35 (2022), 30150–30166.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. 2016. Adversarial feature learning. *arXiv preprint arXiv:1605.09782* (2016).

Kai-Tai Fang and Chang-Xing Ma. 2001. Wrap-around L2-discrepancy of random sampling, Latin hypercube and uniform designs. *Journal of complexity* 17, 4 (2001), 608–624.

Cecile Fauconnier and Gentiane Haesbroeck. 2009. Outliers detection with the minimum covariance determinant estimator in practice. *Statistical Methodology* 6, 4 (2009), 363–379.

Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. 2021. Deep learning for medical anomaly detection–a survey. *ACM Computing Surveys (CSUR)* 54, 7 (2021), 1–37.

Milton Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association* 32, 200 (1937), 675–701.

Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information sciences* 180, 10 (2010), 2044–2064.

Izhak Golan and Ran El-Yaniv. 2018. Deep anomaly detection using geometric transformations. *Advances in neural information processing systems* 31 (2018).

Markus Goldstein and Andreas Dengel. 2012. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track* 1 (2012), 59–63.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. arXiv:1406.2661 [stat.ML] `https://arxiv.org/abs/1406.2661`

Adam Goodge, Bryan Hooi, See-Kiong Ng, and Wee Siong Ng. 2022. Lunar: Unifying local outlier detection methods via graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 6737–6745.

Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. 2020. DROCC: Deep robust one-class classification. In *International conference on machine learning*. PMLR, 3711–3721.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. 2018. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367* (2018).

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. 2022. Adbench: Anomaly detection benchmark. *Advances in neural information processing systems* 35 (2022), 32142–32159.

Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. 2019. Extended isolation forest. *IEEE transactions on knowledge and data engineering* 33, 4 (2019), 1479–1489.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

Zengyou He, Xiaofei Xu, and Shengchun Deng. 2003. Discovering cluster-based local outliers. *Pattern recognition letters* 24, 9-10 (2003), 1641–1650.

Waleed Hilal, S Andrew Gadsden, and John Yawney. 2022. Financial fraud: a review of anomaly detection techniques and recent advances. *Expert systems With applications* 193 (2022), 116429.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

Heiko Hoffmann. 2007. Kernel PCA for novelty detection. *Pattern recognition* 40, 3 (2007), 863–874.

Kyle Hundman, Vasileios Constantinou, Cody Laporte, Ian Colwell, and Tarek Soderstrom. 2018. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 387–395.

Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. 2021. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080* (2021).

Samira Khodabandehlou and Alireza Hashemi Golpayegani. 2024. FiFrauD: unsupervised financial fraud detection in dynamic graph streams. *ACM Transactions on Knowledge Discovery from Data* 18, 5 (2024), 1–29.

Diederik P Kingma, Max Welling, et al. 2013. Auto-encoding variational bayes.

Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. 2008. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 444–452.

Guokun Lai, Yiming Zhao, Wei-Cheng Chang, Yiming Yang, and Eric P Xing. 2021. Revisiting time series outlier detection: Robust prediction and interpretable attribution. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Longin Jan Latecki, Aleksandar Lazarevic, and Dragoljub Pokrajac. 2007. Outlier detection with kernel density functions. In *International workshop on machine learning and data mining in pattern recognition*. Springer, 61–75.

Aleksandar Lazarevic and Vipin Kumar. 2005. Feature bagging for outlier detection. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. 157–166.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. 2006. A tutorial on energy-based learning. *Predicting structured data* 1, 0 (2006).

Sangyun Lee, Beomsu Kim, and Jong Chul Ye. 2023. Minimizing trajectory curvature of ode-based generative models. In *International Conference on Machine Learning*. PMLR, 18957–18973.

Zhong Li, Qi Huang, Lincen Yang, Jiayang Shi, Zhao Yang, Niki van Stein, Thomas Bäck, and Matthijs van Leeuwen. 2025a. Diffusion Models for Tabular Data: Challenges, Current Progress, and Future Directions. *arXiv preprint arXiv:2502.17119* (2025).

Zhong Li, Sheng Liang, Jiayang Shi, and Matthijs van Leeuwen. 2024a. Cross-domain graph level anomaly detection. *IEEE Transactions on Knowledge and Data Engineering* (2024).

Zhong Li, Jiayang Shi, and Matthijs Van Leeuwen. 2024b. Graph neural networks based log anomaly detection and explanation. In *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings*. 306–307.

Zhong Li and Matthijs van Leeuwen. 2022. Feature selection for fault detection and prediction based on event log analysis. *ACM SIGKDD Explorations Newsletter* 24, 2 (2022), 96–104.

Zhong Li, Yuhang Wang, and Matthijs van Leeuwen. 2025b. Towards automated self-supervised learning for truly unsupervised graph anomaly detection. *Data Mining and Knowledge Discovery* 39, 5 (2025), 1–43.

Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H Chen. 2022. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering* 35, 12 (2022), 12181–12193.

Zhong Li, Yuxuan Zhu, and Matthijs Van Leeuwen. 2023. A survey on explainable anomaly detection. *ACM Transactions on Knowledge Discovery from Data* 18, 1 (2023), 1–54.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2022. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747* (2022).

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 eighth ieee international conference on data mining*. IEEE, 413–422.

Jiaqi Liu, Guoyang Xie, Jinbao Wang, Shangnian Li, Chengjie Wang, Feng Zheng, and Yaochu Jin. 2024. Deep industrial image anomaly detection: A survey. *Machine Intelligence Research* 21, 1 (2024), 104–135.

Xingchao Liu, Chengyue Gong, and Qiang Liu. 2022. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003* (2022).

Yezheng Liu, Zhe Li, Chong Zhou, Yuanchun Jiang, Jianshan Sun, Meng Wang, and Xiangnan He. 2019. Generative adversarial active learning for unsupervised outlier detection. *IEEE Transactions on Knowledge and Data Engineering* 32, 8 (2019), 1517–1528.

Victor Livernoche, Vineet Jain, Yashar Hezaveh, and Siamak Ravanbakhsh. 2023. On diffusion modeling for anomaly detection. *arXiv preprint arXiv:2305.18593* (2023).

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems* 30 (2017).

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

Julien Marzat, Hélène Piet-Lahanier, Frédéric Damongeot, and Eric Walter. 2012. Model-based fault diagnosis for aerospace systems: a survey. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of aerospace engineering* 226, 10 (2012), 1329–1360.

Łukasz Maziarka, Marek Śmieja, Marcin Sendera, Łukasz Struski, Jacek Tabor, and Przemysław Spurek. 2021. OneFlow: One-class flow for anomaly detection based on a minimal volume region. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 11 (2021), 8508–8519.

Matthew McDermott, Haoran Zhang, Lasse Hansen, Giovanni Angelotti, and Jack Gallifant. 2024. A closer look at auroc and auprc under class imbalance. *Advances in Neural Information Processing Systems* 37 (2024), 44102–44163.

Francis J Murray and Kenneth S Miller. 2013. *Existence theorems for ordinary differential equations*. Courier Corporation.

Anvardh Nanduri and Lance Sherry. 2016. Anomaly detection in aircraft data using Recurrent Neural Networks (RNN). In *2016 Integrated Communications Navigation and Surveillance (ICNS)*. Ieee, 5C2–1.

Peter Bjorn Nemenyi. 1963. *Distribution-free multiple comparisons.* Princeton University.

Minh-Nghia Nguyen and Ngo Anh Vien. 2019. Scalable and Interpretable One-Class SVMs with Deep Learning and Random Fourier Features. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11051)*. Springer International Publishing, 157–173. doi:10.1007/978-3-030-10925-7_10

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. 2021. Deep learning for anomaly detection: A review. *ACM computing surveys (CSUR)* 54, 2 (2021), 1–38.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2021. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research* 22, 57 (2021), 1–64.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

Tomáš Pevnỳ. 2016. Loda: Lightweight on-line detector of anomalies. *Machine Learning* 102 (2016), 275–304.

Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems* 20 (2007).

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. 2000. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 427–438.

Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark DePristo, Joshua Dillon, and Balaji Lakshminarayanan. 2019. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems (NeurIPS)*. 14680–14691.

Danilo Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *International conference on machine learning*. PMLR, 1530–1538.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep one-class classification. In *International conference on machine learning*. PMLR, 4393–4402.

Edin Šabić, David Keeley, Bailey Henderson, and Sara Nannemann. 2021. Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data. *Ai & Society* 36, 1 (2021), 149–158.

Mayu Sakurada and Takehisa Yairi. 2014. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis*. 4–11.

Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. 2017. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In *Information Processing in Medical Imaging*, Marc Niethammer, Martin Styner, Stephen Aylward, Hongtu Zhu, Ipek Oguz, Pew-Thian Yap, and Dinggang Shen (Eds.). Springer International Publishing, Cham, 146–157.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. 1999. Support vector method for novelty detection. *Advances in neural information processing systems* 12 (1999).

Tom Shenkar and Lior Wolf. 2022. Anomaly detection for tabular data with internal contrastive learning. In *International conference on learning representations*.

Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. 2003. A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the IEEE foundations and new directions of data mining workshop*. IEEE Press Piscataway, NJ, USA, 172–179.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456* (2020).

Mahito Sugiyama and Karsten Borgwardt. 2013. Rapid distance-based outlier detection via sampling. *Advances in neural information processing systems* 26 (2013).

Jian Tang, Zhixiang Chen, Ada Wai-Chee Fu, and David W Cheung. 2002. Enhancing effectiveness of outlier detections for low density patterns. In *Advances in knowledge discovery and data mining: 6th Pacific-Asia conference, PAKDD 2002 Taipei, Taiwan, May 6–8, 2002 proceedings 6*. Springer, 535–548.

David MJ Tax and Robert PW Duin. 2004. Support vector data description. *Machine learning* 54 (2004), 45–66.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Hongzuo Xu, Guansong Pang, Yijie Wang, and Yongjun Wang. 2023a. Deep isolation forest for anomaly detection. *IEEE Transactions on Knowledge and Data Engineering* 35, 12 (2023), 12591–12604.

Hongzuo Xu, Yijie Wang, Juhui Wei, Songlei Jian, Yizhou Li, and Ning Liu. 2023b. Fascinating supervisory signals and where to find them: Deep anomaly detection with scale learning. In *International Conference on Machine Learning*. PMLR, 38655–38673.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023. Diffusion models: A comprehensive survey of methods and applications. *Comput. Surveys* 56, 4 (2023), 1–39.

Jiaxin Yin, Yuanyuan Qiao, Zitang Zhou, Xiangchao Wang, and Jie Yang. 2024. Mcm: Masked cell modeling for anomaly detection in tabular data. In *The Twelfth International Conference on Learning Representations*.

Jianbo Yu and Yue Zhang. 2023. Challenges and opportunities of deep learning-based process fault detection and diagnosis: a review. *Neural Computing and Applications* 35, 1 (2023), 211–252.

Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. 2018. Adversarially learned anomaly detection. In *2018 IEEE International conference on data mining (ICDM)*. IEEE, 727–736.

Yue Zhao, Zain Nasrullah, and Zheng Li. 2019. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research* 20, 96 (2019), 1–7. `http://jmlr.org/papers/v20/19-011.html`

Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. 2023. Improved techniques for maximum likelihood estimation for diffusion odes. In *International Conference on Machine Learning*. PMLR, 42363–42389.

Chong Zhou and Randy C Paffenroth. 2017. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 665–674.

Leixin Zhou, Wenxiang Deng, and Xiaodong Wu. 2020. Unsupervised anomaly localization using VAE and beta-VAE. *arXiv preprint arXiv:2005.10686* (2020).

Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state that the proposed anomaly detection algorithm addresses major limitations of existing generative-model-based approaches, including scalability (in both training and inference), explainability, and provability. These claims are supported by theoretical analysis and an extensive empirical benchmark study.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We outline three limitations of this work in Appendix D.6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All propositions in the paper are accompanied by well-motivated and clearly stated assumptions, along with complete and correct proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides a self-contained description of the proposed algorithm and experimental setup, including all necessary details to reproduce the main results. Additionally, we provide open-source implementations via an anonymous online repository to further support reproducibility.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The main experimental results are based on an open-source benchmark (ADBench), and we provide open-source code with sufficient instructions to ensure faithful reproduction.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides extensive details of the experimental setup. Additional specifics are included in the appendix to ensure clarity and reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report statistical significance tests in Appendix D.5 for our main conclusions, and error bars (in terms of standard deviations) are given on each table and figure (when applicable).

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details of the computing resources used for all experiments, including hardware specifications and runtime, are provided in Appendix B.3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm that all aspects of our research comply with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: Although this paper presents fundamental research in anomaly detection methodology for tabular data and does not involve specific applications or deployments that would raise direct societal concerns, we still outline possible impacts of this work in Appendix D.6.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external assets used in this work, including datasets, code, and models, have been properly credited, and their licenses and terms of use have been appropriately respected.

    Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The source code, along with clear usage instructions, will be released under the MIT license to support reproducibility and ease of use.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: This work does not involve research with human subjects; therefore, IRB approval was not required.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Large language models (LLMs) were not used as part of the core methods in this research; their use was limited to minor editing and polishing of text.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix

Table of Contents

A Related Work

A.1 Anomaly Detection Methods

Unsupervised VS Semi-Supervised. Labeled anomalies are often scarce in practice, as they typically correspond to rare and costly events—such as aerospace system crashes (Marzat et al., 2012; Nanduri and Sherry, 2016), faults in industrial systems (Li and van Leeuwen, 2022; Li et al., 2024b), financial fraud (Hilal et al., 2022; Khodabandehlou and Golpayegani, 2024), or critical health incidents (Šabić et al., 2021; Fernando et al., 2021). As a result, anomaly detection is commonly framed as an unsupervised or semi-supervised task. Compared to supervised settings, most unsupervised approaches face a fundamental challenge: the absence of labeled input-output pairs renders standard regression or classification techniques inapplicable (Liu et al., 2022). Anomaly detection inherits this difficulty, requiring alternative methods to detect abnormality without explicit supervision. To address this, many recent deep learning-based anomaly detection approaches adopt a *semi-supervised* setting (or one-class classification), where the training data consists exclusively of normal instances, which are relatively easier to collect. The underlying principle is that a model trained solely on normal data should learn only normal patterns. When evaluated on test data containing both normal and anomalous instances, those deviating from the learned normal patterns are expected to exhibit larger fitting errors (e.g., reconstruction errors (An and Cho, 2015; Zong et al., 2018), prediction residuals (Lai et al., 2021; Hundman et al., 2018), or likelihood-based scores (Zenati et al., 2018; Ren et al., 2019)), leading to higher anomaly scores. Although many works refer to this setup as *unsupervised anomaly detection*, we use the term *semi-supervised anomaly detection* for conceptual clarity and rigor. This setting is widely adopted in recent deep learning-based anomaly detection

studies such as DeepSVDD (Ruff et al., 2018), VAE (An and Cho, 2015), GANomaly (Akcay et al., 2018), ALAD (Zenati et al., 2018).

Shallow vs. Deep. *Shallow methods* refer to classical anomaly detection approaches that do not rely on neural networks. Representative examples include One-Class SVM (OCSVM) (Schölkopf et al., 1999), Support Vector Data Description (SVDD) (Tax and Duin, 2004), Kernel Density Estimation (KDE) (Latecki et al., 2007), Isolation Forest (IForest) (Liu et al., 2008), and distance-based methods such as k-Nearest Neighbors (KNN) (Angiulli and Pizzuti, 2002), Local Outlier Factor (LOF) (Breunig et al., 2000), and Connectivity-based Outlier Factor (COF) (Tang et al., 2002). Kernel-based methods often suffer from poor scalability, as they require constructing large kernel matrices and storing support vectors during inference (Ruff et al., 2018). IForest, while efficient in low dimensions, tends to degrade in high-dimensional settings due to its reliance on random projections and axis-aligned splits, which may fail to capture meaningful structure in complex data. Similarly, KNN-based methods are sensitive to the curse of dimensionality, where distance metrics lose discriminative power, and their computational complexity scales poorly with dataset size. Overall, these limitations have motivated the development of deep learning-based anomaly detection methods, which can better handle large-scale and high-dimensional data by learning expressive representations.

Deep learning-based anomaly detection methods can be broadly categorized into two groups Ruff et al. (2018): (1) *two-stage approaches*, which utilize neural networks to learn feature representations, followed by classical anomaly detection algorithms applied to these representations; and (2) *end-to-end trained approaches*, which integrate representation learning and anomaly detection objectives within a unified deep learning framework. This paper focuses on the latter category, which has received increasing attention due to its potential for end-to-end optimization and adaptability to complex data modalities. We structure our review around four major lines of *end-to-end trained* approaches: (i) one-class classification paradigms (e.g., Deep SVDD), (ii) generative based models (e.g., GANs-based methods, VAEs-based methods, diffusion models-based methods),(iii) reconstruction-based methods (e.g., autoencoders), and (iv) emerging variants including self-supervised based methods, graph-based methods, and hybrid models. For each group, we review some representative methods and highlight their respective limitations in the following.

A.1.1 One-Class Classification Methods

This line of work draws inspiration from classical one-class classification, such as Support Vector Data Description (SVDD) (Tax and Duin, 2004). For instance, Deep SVDD (Ruff et al., 2018) and its variants train a neural network to map normal data into a compact region in latent space, typically minimizing the distance to a center point or hypersphere. Anomalies are then identified based on their deviation from this learned region. Other examples include AE-1SVM (Nguyen and Vien, 2019), Deep Robust One-Class Classification (DROCC) (Goyal et al., 2020), OneFlow (Maziarka et al., 2021), and they will be reviewed in more details as follows.

DeepSVDD (Ruff et al., 2018). They train a neural network to learn a hypersphere of minimum volume to enclose the embeddings of normal instances, while the embeddings of abnormal instances tend to lie outside the hypersphere. However, it may suffer from the problem of hypersphere collapse, where the hypersphere collapses to a single point. To alleviate this collapse problem, they authors propose that: 1) all-zero-weights solution cannot be used for the center of hypersphere, 2) only unbounded activations should be used, and 3) bias terms need to be omitted, which may lead to sub-optimal feature representation as bias terms are mandatory to shift activation values in neural networks. In contrast, our method TCCM deliberately learns to flow to such a “collapsed" single point, without suffering from any model collapse problem. Moreover, we operate on the input space without learning an explicit latent space, maintaining the explainability of anomaly scores.

AE-1SVM (Nguyen and Vien, 2019). They combine autoencoder (for representation learning) with OCSVM (for anomaly detection) in an end-to-end training manner. Moreover, they extend gradient-based attribution methods to analyze the contribution of input features on anomaly scores. Particularly, to solve the scalability issues with kernel machines (which has a complexity of $O(N^2)$ with N the number of samples) in the original SVM, they employ random Fourier features (Rahimi and Recht, 2007) to approximate the kernel function. They also point out that “the biggest issue of OCSVM is their (poor) capability to handle large and high-dimensional datatsets due to optimization complexity."

DROCC (Goyal et al., 2020). They assume that instances from the normal class lie on a locally linear low dimensional manifold, which is well-sampled in the training data. A test instance is considered as anomalous if it is outside the union of small l_2 balls around the typical normal instances. Particularly, they convert the anomaly detection problem from an unsupervised learning setting to supervised setting as follows: they first generate synthetic anomalous instances (based on the above assumption) to add into the training set, and then train a supervised classifier to distinguish the embeddings of synthetical anomalous instances and those of typical normal instances. This method is robust to representation collapse as mapping all instances into a single point will lead to poor classification results. This method is applicable to tabular data, image, time series, audio, etc. DROCC's optimization is formulated as a saddle point problem, which is solved using standard gradient descent-ascent algorithm (which may be unstable like in adversarial training).

OneFlow (Maziarka et al., 2021). They introduce a one-class anomaly detection method based on NICE flows (Dinh et al., 2014), a type of normalizing flow with a volume-preserving transformation (i.e., constant Jacobian determinant). The core idea is to learn an invertible transformation that maps nominal data to a latent space, and then fit a minimal-volume hypersphere that encloses a fixed proportion of the mapped points. Anomalies are detected based on their distance from the hypersphere center. This approach avoids full density estimation and focuses on directly modeling the support of normal data. A Bernstein polynomial estimator is used to ensure smooth quantile estimation, and the training loss only depends on boundary-adjacent points—resembling support vector behavior. While effective in modeling low-density support, OneFlow has several limitations compared to flow matching methods: the use of volume-preserving flows like NICE limits the expressivity of the learned transformation and the capacity to model complex data distributions.

A.1.2 Generative based Approaches

Generative models, particularly those based on Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), have been widely adapted for anomaly detection by learning to approximate the distribution of normal data (Schlegl et al., 2017; Akcay et al., 2018). These methods typically detect anomalies by measuring reconstruction error, discriminator scores, or deviations in latent space, leveraging GANs' capacity to generate realistic high-dimensional samples. Variational Autoencoders (VAEs) (Kingma et al., 2013) offer an alternative probabilistic formulation, modeling normality via reconstruction likelihood and latent priors. More recently, denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020) have been applied to anomaly detection, often by reconstructing inputs through reverse diffusion trajectories and using the reconstruction residual as an anomaly score (Livernoche et al., 2023). However, such models often suffer from high inference latency due to the iterative nature of reverse sampling. Most notably, a newer class of generative models known as *flow matching* (Lipman et al., 2022; Albergo and Vanden-Eijnden, 2023; Liu et al., 2022) has recently emerged as a powerful and stable alternative to diffusion models. Despite its strong theoretical foundation and demonstrated success in generative modeling, flow matching has not yet been systematically explored for anomaly detection, especially in the tabular setting. This leaves a promising research gap, motivating our development of a flow matching-inspired framework tailored specifically to the needs of semi-supervised anomaly detection.

AnoGAN (Schlegl et al., 2017). This is the first work to employ GANs for anomaly detection. Specifically, they first utilize only normal instances to train a GAN model (including a generator and a discriminator). At test time, given a query image $\boldsymbol{x}$, they iteratively search for a latent embedding $\boldsymbol{z}$ such that the generated image $G(\boldsymbol{z})$ closely resembles $\boldsymbol{x}$ and lies on the learned data manifold. This is done by minimizing a joint loss consisting of a residual loss and a discrimination loss through Γ steps of backpropagation. However, such per-instance optimization leads to a high inference cost, making the method less practical for real-time applications. To quantify the abnormality of a test sample, AnoGAN defines an *anomaly score* as a weighted sum of the residual loss and the discrimination loss obtained after Γ optimization steps in the latent space. Specifically, the residual loss captures the pixel-wise difference between the input image $\boldsymbol{x}$ and the generated image $G(\boldsymbol{z}_\Gamma)$, while the discrimination loss measures how well the generated image fits on the learned data manifold. A high anomaly score indicates poor reconstruction and/or low likelihood under the discriminator, both of which suggest the input is dissimilar to the normal training distribution. In addition to scalar scoring, AnoGAN also provides visual explanation by computing a residual image $\boldsymbol{x}_R = |\boldsymbol{x} - G(\boldsymbol{z}_\Gamma)|$, highlighting regions that contribute most to the anomaly.

ALAD (Zenati et al., 2018). They propose Adversarially Learned Anomaly Detection (ALAD), a reconstruction-based GAN framework tailored for efficient unsupervised anomaly detection. Unlike earlier methods such as AnoGAN that require costly per-instance optimization during inference, ALAD incorporates an encoder network that directly maps inputs to the latent space, enabling fast, one-pass anomaly scoring. The model builds upon the BiGAN framework (Donahue et al., 2016) by jointly training a generator, discriminator, and encoder, with additional cycle-consistency regularizations to enforce accurate reconstruction. Specifically, it introduces two auxiliary discriminators to enforce consistency in both data and latent spaces, improving the quality of reconstructions. To detect anomalies, ALAD defines an anomaly score based on the feature difference between the input and its reconstruction, extracted from an intermediate layer of the discriminator operating on sample pairs. This feature-level distance serves as a more robust indicator than raw pixel differences, especially for high-dimensional data. While ALAD achieves fast inference and strong performance, it relies on adversarially balancing multiple networks and loss components during training, which may introduce stability challenges and increased complexity compared to simpler reconstruction-based approaches.

GANomaly (Akcay et al., 2018). It is a representative GAN-based anomaly detection framework that enhances vanilla GANs by incorporating an explicit encoder–decoder structure. Instead of relying on latent space search at test time, GANomaly introduces a dual-encoder architecture to enable efficient, feedforward inference. During training, only normal data are used to train a generator network composed of an encoder and decoder ($G = G_D \circ G_E$), which learns to reconstruct the input images. In parallel, a second encoder E is trained to map the reconstructed images back into the latent space. The key assumption is that, for normal samples, the original and reconstructed latent vectors ($z = G_E(x)$ and $\hat{z} = E(G(x))$) should be close, whereas for anomalous inputs, their discrepancy will be larger. The training objective combines three losses: (i) a contextual loss that encourages pixel-wise similarity between input and reconstruction, (ii) a feature matching loss computed from intermediate discriminator activations to stabilize adversarial learning, and (iii) a latent encoder loss that penalizes the difference between z and $\hat{z}$. At test time, the anomaly score is defined based solely on the latent distance $\mathcal{A}(x) = |G_E(x) - E(G(x))|_1$, allowing for fast, one-pass anomaly detection without the iterative inference used in methods like AnoGAN. While GANomaly achieves a good trade-off between accuracy and efficiency, it has two notable limitations. First, its anomaly score depends entirely on the latent discrepancy, which may be vulnerable to representation collapse or ambiguous reconstructions. Second, the three-part loss requires balancing multiple objectives, and tuning the corresponding weights without labeled data can be challenging in unsupervised settings.

SO-GAAL and **MO-GAAL** (Liu et al., 2019) . They propose two GAN-based outlier detection methods that approach anomaly detection as an adversarial learning process. In *Single-Objective Generative Adversarial Active Learning* (SO-GAAL), a generator is trained to synthesize informative potential outliers, while a discriminator attempts to distinguish these from the real data. This adversarial interplay gradually improves the quality of the generated outliers, enabling the discriminator to carve a tighter decision boundary around normal data. However, SO-GAAL may suffer from mode collapse and performance degradation once the generator overfits the data manifold. To address this, the authors further propose *Multiple-Objective GAAL* (MO-GAAL), which introduces multiple sub-generators, each responsible for generating outliers relative to a specific subset of the data. By doing so, MO-GAAL builds a more diverse and comprehensive reference distribution, allowing the discriminator to maintain stable and accurate detection even when dealing with multi-modal or high-dimensional data. Both approaches ultimately compute an outlier score based on the discriminator's output, with higher scores indicating greater deviation from the learned normal distribution.

VAE (Kingma et al., 2013). They introduce a principled probabilistic framework for learning latent representations via a variational autoencoder. For anomaly detection, the VAE is trained solely on normal instances, learning to encode data into a low-dimensional latent space and reconstruct inputs through a decoder (An and Cho, 2015; Zhou et al., 2020). During training, the model jointly minimizes a reconstruction loss and a KL divergence regularizer, which encourages the latent codes to follow a standard Gaussian prior. At test time, given a new input $\boldsymbol{x}$, the model produces a reconstructed sample $\hat{\boldsymbol{x}}$ by encoding and decoding it through the learned latent space. The anomaly score is then computed based on the reconstruction error, typically using an ℓ_2 distance between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$. Since the model is trained to reconstruct normal patterns well, higher reconstruction errors suggest that the input deviates from the normal data distribution. Compared to GAN-based approaches like AnoGAN, VAE offers faster inference as no iterative optimization is required at test time, making it more practical for real-time applications. However, the generative quality of

VAEs is often inferior to GANs, especially when dealing with complex or high-dimensional data. In some cases, even anomalous inputs can be reconstructed with low error, leading to false negatives. Furthermore, the balance between reconstruction fidelity and latent regularization (e.g., via the KL term or the β coefficient in β-VAE) is sensitive, and improper tuning may lead to posterior collapse or over-regularization, which degrades anomaly detection performance.

DTE (Livernoche et al., 2023). They propose a novel use of diffusion processes for anomaly detection by predicting the noise level—or diffusion timestep—associated with an input sample. The core intuition is that normal instances lie close to the data manifold and hence resemble samples with low diffusion noise, while anomalies lie further away and mimic samples diffused with stronger noise. During training, DTE simulates noisy samples via a predefined forward diffusion process (e.g., variance-preserving), and learns a neural network to regress or classify the corresponding diffusion time, using only normal data. At test time, inputs that are harder to explain as low-noise samples receive higher predicted diffusion times and are flagged as anomalies. Notably, this approach avoids learning the full reverse process as in DDPMs and instead frames anomaly detection as a diffusion time estimation task. However, DTE—particularly its non-parametric variant—faces major scalability bottlenecks. DTE-NonParametric estimates the diffusion time of a test sample by computing a posterior over all training points, using distance-based kernel density approximations. This requires comparing each test input against a large training set, making inference prohibitively slow for high-volume or high-dimensional data. Moreover, the diffusion process used to synthesize training data adds to the overall preprocessing overhead, limiting DTE's suitability for real-time or resource-constrained settings. Despite its strong detection performance, these computational costs hinder its broad applicability in large-scale anomaly detection pipelines.

A.1.3 Reconstruction-based Methods

Reconstruction-based methods constitute a major paradigm in anomaly detection. The central idea is that models trained to accurately reconstruct normal data will fail to do so for anomalous inputs, which typically deviate from the learned data manifold. The reconstruction error—measured in input space or latent space—is then used as an anomaly score. Among the most widely used reconstruction-based models are Autoencoders (AEs) and their variants (Sakurada and Yairi, 2014; Zhou and Paffenroth, 2017). These models learn compact representations through a bottleneck architecture and are trained to minimize the reconstruction loss on normal data. Anomalies are expected to produce larger reconstruction errors due to their poor alignment with the learned representation space. Variational Autoencoders (VAEs) (An and Cho, 2015) further extend this idea by introducing a probabilistic latent space, allowing uncertainty-aware reconstructions.

Beyond classical AEs, many generative models also incorporate reconstruction-based objectives. For example, GAN-based methods such as GANomaly (Akcay et al., 2018) and ALAD (Zenati et al., 2018) utilize an encoder–decoder–discriminator pipeline, where anomaly scores are derived from reconstruction fidelity or latent-space consistency. Similarly, recent diffusion-based methods (Livernoche et al., 2023) detect anomalies by reconstructing inputs through reverse diffusion trajectories. As other methods have been (or will be) reviewed in other parts, we will review a representative reconstruction-based approach, DAGMM (Zong et al., 2018), in the following.

DAGMM (Zong et al., 2018). They propose the Deep Autoencoding Gaussian Mixture Model (DAGMM), a unified deep framework for unsupervised anomaly detection that jointly learns low-dimensional representations and density estimation. Specifically, DAGMM integrates two key components: a compression network, which is a deep autoencoder producing both latent features and reconstruction error metrics; and an estimation network, which models a Gaussian Mixture Model (GMM) over the concatenated features to estimate sample energy (i.e., negative log-likelihood). To avoid traditional two-step training, DAGMM jointly optimizes the autoencoder and the GMM via a shared objective that includes reconstruction loss, sample energy, and a regularization term to prevent degenerate covariance matrices. During inference, an anomaly score is computed as the energy of a test sample under the learned GMM, with higher energy indicating greater anomaly likelihood. Unlike conventional approaches that rely only on reconstruction error or pre-trained representations, DAGMM is trained end-to-end, enabling the autoencoder to adapt its compression strategy in favor of improved density estimation. This results in enhanced ability to detect subtle or "lurking" anomalies that might not have high reconstruction errors but reside in low-density regions. One limitation of DAGMM is that its network configuration—such as the number of mixture components in the GMM, and the architectures of the compression and estimation networks—needs to be selected in a

data-dependent manner. These choices often require dataset-specific tuning, which may affect the model's ease of deployment and generalizability across tasks.

Limitations of Autoencoders. Autoencoders are typically trained to reconstruct the input data while enforcing an intermediate low-dimensional representation, which acts as an information bottleneck to encourage the neural network to extract salient features from the training data. In the context of semi-supervised anomaly detection, this promotes learning the underlying factors of variation shared among normal instances. However, since autoencoders do not directly optimize for anomaly detection, their effectiveness heavily depends on how well the latent space captures relevant structure in the data. In particular, the choice of latent dimensionality becomes critical: if it is too high, the model may simply memorize the input; if it is too low, essential information may be lost. This hyperparameter is often data-dependent and difficult to tune due to the unsupervised nature of the task and the challenges in estimating the intrinsic dimensionality of the data (Bengio et al., 2013).

A.1.4 Self-Supervised based Methods and Other Miscellaneous Methods

Recent studies explore alternative deep paradigms for anomaly detection, including self-supervised learning-based methods such as GOAD (Bergman and Hoshen, 2020), ICL (Shenkar and Wolf, 2022), SLAD (Xu et al., 2023b), and MCM (Yin et al., 2024), graph neural network-based method such as LUNAR (Goodge et al., 2022), and hybrid models such as DIF (Xu et al., 2023a) that combine deep feature learning with traditional detectors. They will be reviewed in more detail as follows.

GOAD (Bergman and Hoshen, 2020). They introduce a classification-based approach for anomaly detection that unifies one-class and transformation-based paradigms. It first applies a set of M geometric or affine transformations to each normal training instance and learns a shared feature extractor f that maps transformed inputs to a representation space. Each transformed variant is encouraged to cluster around a distinct center using a triplet-style loss, promoting intra-class compactness and inter-class separation. At inference, GOAD computes the transformation prediction likelihood for each transformed test instance and aggregates these into a final anomaly score: samples that are poorly aligned with any learned transformation subspace are considered anomalous. Unlike earlier transformation-based methods (e.g., GEOM (Golan and El-Yaniv, 2018)) that suffer from unreliable extrapolation to unseen anomalies, GOAD regularizes prediction confidence on out-of-distribution regions and generalizes to non-image data via learnable affine transformations. However, GOAD's effectiveness heavily depends on the quality and diversity of the transformations used, and its performance is sensitive to the choice of the number of transformations M, which must be manually specified and tuned per dataset—potentially limiting its practicability across domains.

ICL (Shenkar and Wolf, 2022). They introduce a novel approach to anomaly detection in tabular data by leveraging contrastive learning on internal feature partitions. Unlike methods that depend on external transformations or assume data structure (e.g., spatial correlations in images), ICL operates under the premise that dependencies among feature subsets are class-specific. Specifically, given a single-class training set, the method slides a window of size k over each input vector $\boldsymbol{x} \in \mathbb{R}^d$ to generate a set of $m = d - k + 1$ paired sub-vectors $(\boldsymbol{a}_j, \boldsymbol{b}_j)$, where $\boldsymbol{a}_j$ is a segment of k consecutive features and $\boldsymbol{b}_j$ is its complement. Two neural networks G and F embed $\boldsymbol{a}_j$ and $\boldsymbol{b}_j$, respectively, into a shared latent space $\mathbb{R}^u$, trained to maximize mutual information between matching pairs via a noise contrastive loss. At test time, the anomaly score for a sample $\boldsymbol{x}$ is defined as the sum of contrastive losses across all j, directly measuring how class-consistent its internal structure is under the learned embeddings. Importantly, the method is interpretable by design: the local loss for each feature subset allows pinpointing which attributes contribute most to an anomaly. While hyperparameters such as the window size k and latent dimension u must be set (in a data-dependent way), empirical evidence shows that performance is robust across a wide range of values, and no dataset-specific tuning is required.

SLAD (Xu et al., 2023b). They propose Scale Learning-based Anomaly Detection (SLAD), a self-supervised framework for tabular anomaly detection that avoids reliance on reconstruction losses. SLAD introduces a novel supervision signal—scale—which quantifies the relationship between subspace dimensionality and representation complexity. Specifically, it samples subspaces of each input, maps them to fixed-length representations, and defines scale labels to supervise the learning of a ranking function via distribution alignment. During inference, anomaly scores are computed by measuring the divergence between predicted and target scale distributions, based on the assumption that normal samples produce more consistent and predictable scales. Despite its conceptual novelty

and strong empirical performance, SLAD presents several limitations. First, the generation of scale labels and the assumption that anomalies inherently exhibit scale inconsistencies may not hold uniformly across datasets or domains, particularly when anomalies are subtle or lie near the decision boundary. Second, SLAD's reliance on distribution alignment adds algorithmic complexity and hyperparameter sensitivity, which may impact robustness in practical deployment. Finally, while the model is self-supervised, its interpretability remains limited, as the learned scale concept is abstract and may not directly correspond to human-understandable explanations.

MCM (Yin et al., 2024). This is a novel self-supervised framework for anomaly detection in tabular data. Inspired by masked modeling techniques in NLP and vision (e.g., BERT and MAE), MCM learns to reconstruct randomly masked subsets of input features using only the unmasked portions. The core hypothesis is that normal instances exhibit strong internal feature correlations, which the model can learn to reconstruct effectively—while anomalies violate such correlations, leading to higher reconstruction error. To enhance robustness, MCM introduces a *learnable masking strategy* that dynamically generates multiple soft masks per instance. These masks are trained end-to-end via a mask generator network. A *diversity loss* is used to ensure that different masks capture complementary correlations among features, thereby improving the model's discriminative power. The final anomaly score is computed as the average reconstruction error across all masked versions. Compared to prior methods such as contrastive learning (e.g., ICL), which often rely on engineered transformations, MCM provides a more data-driven and flexible mechanism to capture normality. Moreover, the ensemble of masks makes the method more expressive while maintaining a lightweight architecture based on an encoder-decoder MLP. MCM also offers interpretability through both per-mask and per-feature contributions, facilitating insight into which correlations are violated by a given anomaly.

LUNAR (Goodge et al., 2022). They propose a local outlier detection framework based on graph neural networks (GNNs) with learnable message aggregation. Instead of relying on raw feature vectors, LUNAR constructs a k-nearest neighbor graph from the training data and uses pairwise distances as edge features. A single-layer GNN is trained to distinguish normal points from synthetic negative samples by aggregating the distance vector from each node's neighborhood. This design enables the model to learn a parametric anomaly scoring function that adapts better than classical non-trainable local methods such as LOF or KNN. Negative samples are generated using a mix of uniform sampling and feature-space perturbation, which prevents the model from collapsing to trivial solutions. While LUNAR demonstrates strong empirical performance, its reliance on k-NN graph construction introduces scalability challenges in high-dimensional settings, where distance metrics become less meaningful. Moreover, because it avoids using raw features and instead encodes distance information through a learnable aggregation function, it can incur additional computational cost in preprocessing and training—especially on large-scale datasets with many neighbors per node. As the nearest neighbor graph must be recomputed for each new input distribution and all neighborhood distances are fed into the model, the method may be less suitable for real-time or streaming applications compared to embedding-based approaches.

DIF (Xu et al., 2023a). They propose the *Deep Isolation Forest* (DIF), a scalable anomaly detection method that extends isolation-based techniques by incorporating random deep representations. Instead of applying axis-aligned splits on raw features (as in iForest (Liu et al., 2008)), DIF first transforms input data into multiple representation spaces using randomly initialized, optimization-free neural networks. These transformations enable nonlinear partitioning in the original space via simple axis-parallel cuts in the projected space. To ensure efficiency, DIF introduces a computation-efficient ensemble mechanism (CERE) that allows all ensemble members to be computed simultaneously in a mini-batch. For scoring, DIF further proposes a deviation-enhanced anomaly scoring function (DEAS) that combines traditional path length with deviation from split thresholds to reflect local density and isolation difficulty. The authors show that DIF generalizes both iForest and Extended Isolation Forest (EIF) (Hariri et al., 2019), while preserving linear scalability and offering stronger expressive power. However, DIF relies heavily on the randomness and diversity of representations for performance, which may lead to instability without sufficient ensemble size or structure-aware initialization.

A.2 Generative Models

Generative models span a broad spectrum of paradigms, including energy-based models (LeCun et al., 2006), variational autoencoders (VAEs) (An and Cho, 2015), generative adversarial networks

(GANs) (Goodfellow et al., 2014), normalizing flows (Papamakarios et al., 2021), autoregressive models (e.g., Transformers (Vaswani et al., 2017)), diffusion models (Ho et al., 2020), and the more recent flow matching framework (Lipman et al., 2022; Albergo and Vanden-Eijnden, 2023; Liu et al., 2022). While each of these approaches offers unique modeling capabilities, we focus our discussion on flow matching, as it is most directly relevant to the methodology developed in this paper.

GANs and Diffusion Models. Generative adversarial networks (GANs) (Goodfellow et al., 2014) have long been the de facto choice for high-fidelity image generation, but diffusion models have recently surpassed them in terms of mode coverage and conditional flexibility. Despite their effectiveness, diffusion models (Yang et al., 2023) are notoriously computationally intensive, often requiring hundreds of iterative denoising steps to generate a single sample. This has spurred efforts to accelerate training and inference (Dockhorn et al., 2022; Jolicoeur-Martineau et al., 2021); however, many of these approaches still suffer from slow convergence and rely on carefully constructed probability paths, which limit scalability on high-dimensional or large-scale datasets (Dao et al., 2023). Particularly, there is a recent survey paper on diffusion models for tabular data (Li et al., 2025a), which systematically reviewed existing diffusion models for tabular data modeling (including anomaly detection).

Normalizing Flows and Continuous Normalizing Flows. Continuous normalizing flows (CNFs) (Chen et al., 2018; Grathwohl et al., 2018) model invertible transformations between distributions using neural ordinary differential equations (ODEs). While theoretically elegant, training such models is computationally intensive, as it involves solving ODEs during each forward and backward pass, making scalability a major bottleneck. To overcome this, recent works on *flow matching* (Albergo and Vanden-Eijnden, 2023; Lipman et al., 2022; Liu et al., 2022) propose simulation-free alternatives that avoid explicit trajectory integration. Inspired by ideas from score-based diffusion models, these methods enable more efficient training of CNFs by directly learning velocity fields without solving full ODE systems.

Flow Matching. Flow matching has emerged as a promising alternative that inherits many of the desirable properties of diffusion models—such as robustness and expressivity—while avoiding their main drawbacks. Rather than relying on stochastic differential equations (SDEs), flow matching learns an ordinary differential equation (ODE) that deterministically maps samples from a source distribution ρ_0 (ρ_{source}) to a target distribution ρ_1 (ρ_{target}). This shift from stochastic to deterministic dynamics leads to lower curvature in generative trajectories, which translates into improved stability, faster sampling, and easier optimization (Liu et al., 2022; Lee et al., 2023). The simplicity of its training framework also facilitates broader adoption in settings where computational efficiency is critical. Beyond efficiency, flow matching offers notable modeling flexibility and interpretability. It eliminates the need for a forward diffusion process, and training can be performed by directly matching vector fields between arbitrary distributions (CSAIL, 2024; Albergo and Vanden-Eijnden, 2023). This generality stands in contrast to denoising diffusion models, which often assume Gaussian base distributions and Gaussian interpolants. In addition, the interpolant formulation of flow matching allows evaluation of intermediate densities ρ_t at arbitrary time points $t \in [0, 1]$, enabling empirical inspection of the learned velocity field throughout the trajectory (Albergo and Vanden-Eijnden, 2023). Although this capability may not always be required—e.g., in anomaly detection we often focus only on the terminal velocity at $t = 1$—it enhances interpretability. Lastly, because flow matching only requires samples from the source and target distributions (not their explicit densities), it is particularly well-suited to scenarios with implicit or intractable data distributions.

Difference from Generative Models. While the main objective of normalizing flows (Rezende and Mohamed, 2015) or flow matching (Lipman et al., 2022) is to transform a simple initial distribution into a complex, often multimodal, target distribution—either through a sequence of invertible mappings (in the case of normalizing flows) or via velocity fields (in flow matching)—our focus, by contrast, is on measuring the distance to the target distribution after applying the learned one step velocity field (at any given time). Importantly, this target distribution is a degenerate distribution, which stands in stark contrast to the typical emphasis in generative modeling on the validity and richness of the target distribution.

B Experiment setups

B.1 Datasets

A summary of the datasets used in our study is provided in Table 1. We adopt 47 benchmark datasets from the well-established ADBENCH benchmark (Han et al., 2022), spanning diverse domains including sociology, finance, linguistics, physics, and healthcare. To enable a comprehensive evaluation of different anomaly detectors, including our proposed method, we categorize the datasets into four groups based on their scale and dimensionality: (a) *High-dimensional* datasets, with more than 50 features; (b) *Large-scale* datasets (but not high-dimensional), containing more than 10,000 instances and fewer than 50 features; (c) *Medium-scale* datasets (not high-dimensional), with 1,000 to 10,000 instances; and (d) *Small-scale* datasets (not high-dimensional), containing fewer than 1,000 instances. This categorization facilitates a nuanced analysis of model performance across varying data regimes.

B.2 Baselines

Before introducing the baseline methods, we clarify an important distinction in anomaly detection paradigms: **inductive** vs. **transductive** approaches. Inductive methods learn a generalizable decision function from the training set and apply it directly to unseen test data. In contrast, transductive methods rely on the distribution of the test set during inference, often scoring anomalies relative to the entire evaluation batch. While inductive approaches are generally preferred in deployment scenarios where test data is unavailable during training, transductive methods are still commonly included in benchmarking for historical and comparative purposes.

Our proposed method TCCM is an inductive method, as it learns a model using training data and then computes anomaly scores with the unseen test data. Although comparing inductive and transductive methods directly may not always be ideal due to differing assumptions, we include both for completeness. We categorize the anomaly detection baselines into two main groups:

- 21 Classical Machine Learning-based Methods (Transductive and Inductive). (1) Transductive Methods: **ABOD** (Kriegel et al., 2008), **COF** Tang et al. (2002), **LOF** (Breunig et al., 2000), **PCA** (Shyu et al., 2003), **KPCA** (Hoffmann, 2007), **KNN** (Ramaswamy et al., 2000), **INNE** (Bandaragoda et al., 2018); and (2) Inductive Methods: **CBLOF**(He et al., 2003), **CD** (Cook, 1977), **ECOD** (Li et al., 2022) **FeatureBagging** (Lazarevic and Kumar, 2005), **GMM** (Agarwal, 2007), **HBOS** (Goldstein and Dengel, 2012), **IForest** (Liu et al., 2008), **KDE** (Latecki et al., 2007), **LMDD** (Arning et al., 1996), **LODA** (Pevnỳ, 2016), **MCD** (Fauconnier and Haesbroeck, 2009), **OCSVM** (Schölkopf et al., 1999), **QMCD** (Fang and Ma, 2001), and **Sampling** (Sugiyama and Borgwardt, 2013).

- 23 Deep Learning-based Methods (All are inductive). **AutoEncoder** (Sakurada and Yairi, 2014; Aggarwal and Aggarwal, 2017b), **ALAD** (Zenati et al., 2018), **DIF** (Xu et al., 2023a), **DeepSVDD** (Ruff et al., 2018), **LUNAR** (Goodge et al., 2022), **MOGAAL** (Liu et al., 2019), **SOGAAL** (Liu et al., 2019), **VAE** (An and Cho, 2015), **AE-1SVM** (Nguyen and Vien, 2019), **AnoGAN** (Schlegl et al., 2017), **DAGMM** (Zong et al., 2018), **PlanarFlow** (Normalizing Flows) (Rezende and Mohamed, 2015), **SLAD** (Xu et al., 2023b), **MCM** (Yin et al., 2024), **ICL**(Shenkar and Wolf, 2022), **GOAD**(Bergman and Hoshen, 2020), **GANomaly**(Akcay et al., 2018), **DTE-Categorical** (Livernoche et al., 2023), **DTE-Gaussian** (Livernoche et al., 2023), **DTE-InverseGamma** (Livernoche et al., 2023), **DTE-NonParametric** (Livernoche et al., 2023), **DROCC** (Goyal et al., 2020), **Anomaly-DDPM** (Livernoche et al., 2023).

B.3 Configurations

All experiments are independently performed five times with different random seeds (0, 1, 2, 3, and 4) on each dataset for all 44 baselines and our proposed TCCM with high reproducibility to ensure high robustness, and account for variability due to random initialization.

Implementation Details of TCCM[2]. The time-conditioned velocity field $f_\theta(\boldsymbol{x}, t)$ is parameterized by a 3-layer multilayer perceptron (MLP) with 256 hidden units per layer and ReLU activations. To

[2]Code available at: `https://github.com/ZhongLIFR/TCCM-NIPS`

Table 1: Summary of Datasets. To systematically evaluate the performance of various anomaly detectors, we categorize the datasets into four groups based on their data scale and dimensionality: (a) *high-dimensional* datasets, which contain more than 50 features; (b) *large-scale* datasets (but not high-dimensional), with more than 10,000 instances and fewer than 50 features; (c) *medium-scale* datasets (not high-dimensional), with between 1,000 and 10,000 samples; and (d) *small-scale* (not high-dimensional) datasets, consisting of fewer than 1,000 instances. This categorization allows for a nuanced analysis of model behavior under different data regimes.

Dataset	# Samples	# Features	# Anomaly	% Anomaly	Domain	Category
census	299285	500	18568	6.2	Sociology	High-dimensional
backdoor	95329	196	2329	2.44	Network	High-dimensional
campaign	41188	62	4640	11.27	Finance	High-dimensional
mnist	7603	100	700	9.21	Image	High-dimensional
speech	3686	400	61	1.65	Linguistics	High-dimensional
optdigits	5216	64	150	2.88	Image	High-dimensional
SpamBase	4207	57	1679	39.91	Document	High-dimensional
musk	3062	166	97	3.17	Chemistry	High-dimensional
InternetAds	1966	1555	368	18.72	Image	High-dimensional
donors	619326	10	36710	5.93	Sociology	Large
http	567498	3	2211	0.39	Web	Large
cover	286048	10	2747	0.96	Botany	Large
fraud	284807	29	492	0.17	Finance	Large
skin	245057	3	50859	20.75	Image	Large
celeba	202599	39	4547	2.24	Image	Large
smtp	95156	3	30	0.03	Web	Large
ALOI	49534	27	1508	3.04	Image	Large
shuttle	49097	9	3511	7.15	Astronautics	Large
magic.gamma	19020	10	6688	35.16	Physical	Large
mammography	11183	6	260	2.32	Healthcare	Large
annthyroid	7200	6	534	7.42	Healthcare	Medium
pendigits	6870	16	156	2.27	Image	Medium
satellite	6435	36	2036	31.64	Astronautics	Medium
landsat	6435	36	1333	20.71	Astronautics	Medium
satimage-2	5803	36	71	1.22	Astronautics	Medium
PageBlocks	5393	10	510	9.46	Document	Medium
Wilt	4819	5	257	5.33	Botany	Medium
thyroid	3772	6	93	2.47	Healthcare	Medium
Waveform	3443	21	100	2.9	Physics	Medium
Cardiotocography	2114	21	466	22.04	Healthcare	Medium
fault	1941	27	673	34.67	Physical	Medium
cardio	1831	21	176	9.61	Healthcare	Medium
letter	1600	32	100	6.25	Image	Medium
yeast	1484	8	507	34.16	Biology	Medium
vowels	1456	12	50	3.43	Linguistics	Medium
Pima	768	8	268	34.9	Healthcare	Small
breastw	683	9	239	34.99	Healthcare	Small
WDBC	367	30	10	2.72	Healthcare	Small
Ionosphere	351	32	126	35.9	Oryctognosy	Small
Stamps	340	9	31	9.12	Document	Small
vertebral	240	6	30	12.5	Biology	Small
WBC	223	9	10	4.48	Healthcare	Small
glass	214	7	9	4.21	Forensic	Small
WPBC	198	33	47	23.74	Healthcare	Small
Lymphography	148	18	6	4.05	Healthcare	Small
wine	129	13	10	7.75	Chemistry	Small
Hepatitis	80	19	13	16.25	Healthcare	Small

incorporate time information, we apply fixed sinusoidal embeddings (Vaswani et al., 2017) to the scalar input $t \in [0, 1]$, following the positional encoding strategy used in transformer architectures. The time embedding (default dimension: 128) is concatenated with the input vector $\boldsymbol{x}$, and the combined representation is passed through the MLP to produce the predicted flow vector. We use the Adam optimizer with a learning rate of 0.005. The batch size is set to 1024 for datasets with more than 10,000 samples and to $\min(512, \#\text{training instances})$ for smaller datasets. The number of training epochs is determined empirically using the unsupervised hyperparameter selection method proposed by Li et al. (2025b), which requires no access to anomaly labels. While their method supports per-seed tuning, for consistency and fair evaluation, we fix the number of epochs across different random seeds. Notably, thanks to the efficiency of TCCM, tuning this single hyperparameter incurs minimal computational overhead. This is also the only data-dependent hyperparameter in our setup. The choices of key hyperparameters for our TCCM are presented in Table 5.

Implementations of Other Baselines. We utilise the well-established PyOD package (Zhao et al., 2019) for implementing (1) all classical anomaly detectors such as IForest (Liu et al., 2008), KDE (Latecki et al., 2007), etc., and (2) some deep anomaly detectors, including AutoEncoder (Aggarwal and Aggarwal, 2017b), ALAD (Zenati et al., 2018), DIF (Xu et al., 2023a), DeepSVDD (Ruff et al., 2018), LUNAR (Goodge et al., 2022), MOGAAL (Liu et al., 2019), SOGAAL (Liu et al., 2019), VAE (An and Cho, 2015), AE-1SVM (Nguyen and Vien, 2019), and AnoGAN (Schlegl et al., 2017); Additionally, we adapt the implementations from ADBench [3] for DAGMM (Zong et al., 2018) and GANAnomaly (Akcay et al., 2018); Besides, we include various advanced deep detectors, DROCC (Goyal et al., 2020)[4], GOAD (Bergman and Hoshen, 2020)[5], ICL (Shenkar and Wolf, 2022)[6], SLAD (Xu et al., 2023b)[7], MCM (Yin et al., 2024)[8], DTE (with four variants DTE-Categorical, DTE-Gaussian, DTE-InverseGamma, and DTE-NonParametric and a modified DDPM) (Livernoche et al., 2023)[9]; The implementation of planar flows (Rezende and Mohamed, 2015), a normalizing-flows-based detector, is also taken from (Livernoche et al., 2023). For all baseline detectors, we use their default configurations and hyperparameters as provided by their source implementations.

Hardware and Software. All experiments are conducted on machines equipped with Intel Xeon Gold 6430 CPUs (3.4 GHz, same model across runs, though not necessarily the same physical unit) and 256 GB RAM. No GPU acceleration is used. To ensure a fair comparison, each model is restricted to run on a *single* CPU core, allocated up to 10 GB of RAM, and a maximum runtime of 3 days per dataset. Our implementation is based on Python 3.9.21 with PyTorch 2.0, and experiments are executed within a conda-managed environment running Ubuntu 22.04.

B.4 Evaluation metrics

We evaluate our proposed method and baselines using two standard metrics: Area Under the Receiver Operating Characteristic curve (AUROC) and Area Under the Precision-Recall Curve (AUPRC) (McDermott et al., 2024). Both metrics range from 0 to 1, with higher values indicating better performance. AUROC reflects a method's ability to distinguish between normal and anomalous instances: a score near 1 indicates near-perfect performance, 0.5 corresponds to random guessing, and values below 0.5 imply worse-than-random behavior. For AUPRC, which is more informative in imbalanced settings, higher values reflect better precision-recall trade-offs. All experiments are repeated over 5 independent runs with different random seeds, and we report the mean and standard deviation of each metric for every (dataset, anomaly detector) pair. For each dataset, we also compute detector rankings based on their mean AUROC and AUPRC. Due to the scale of the experiments, we present the complete tables of AUROC and AUPRC scores in the appendix. In the main paper, we visualize the distribution of ranks—computed from the mean AUROC and AUPRC scores across 5 runs—over all datasets using box plots. These plots are ordered by overall performance, defined as

[3] ADBench: `https://github.com/Minqi824/ADBench/tree/main/adbench/baseline`

[4] DROCC: `https://github.com/microsoft/EdgeML/blob/master/pytorch/edgeml_pytorch`

[5] GOAD: `https://github.com/lironber/GOAD`

[6] ICL: in the supplementary material of `https://openreview.net/forum?id=_hszZbt46bT`

[7] SLAD: `https://github.com/xuhongzuo/scale-learning`

[8] MCM: `https://github.com/JXYin24/MCM`

[9] DTE & DDPM: `https://github.com/vicliv/DTE`

the average rank across datasets, where each rank is based on the per-dataset mean score aggregated over the 5 runs.

B.5 Pseudo-code of TCCM

Algorithm 1 TCCM Training

Input: Training data samples $\boldsymbol{z} \sim p_{\text{data}}$, neural network $f_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$, number of training epochs N_{epochs}, batch size B, learning rate η.
Initialize model parameters $\boldsymbol{\theta}$.
for epoch = 1 to N_{epochs} **do**
  Shuffle training data.
  for each batch $\{\boldsymbol{z}^{(i)}\}_{i=1}^{B}$ from p_{data} **do**
    Sample time steps $\{t^{(i)}\}_{i=1}^{B}$ where each $t^{(i)} \sim \mathcal{U}(0,1)$.
    Generate time embeddings: $\boldsymbol{e}_t^{(i)} \leftarrow \text{SinusoidalEmbedding}(t^{(i)})$ for $i = 1, \ldots, B$.
    Form augmented inputs: $\tilde{\boldsymbol{z}}^{(i)} \leftarrow [\boldsymbol{z}^{(i)}; \boldsymbol{e}_t^{(i)}]$ for $i = 1, \ldots, B$.
    Predict contraction vectors: $\hat{\boldsymbol{v}}^{(i)} \leftarrow f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{z}}^{(i)})$ for $i = 1, \ldots, B$.
    Compute batch loss: $\mathcal{L}(\boldsymbol{\theta}) \leftarrow \frac{1}{B} \sum_{i=1}^{B} \left\| \hat{\boldsymbol{v}}^{(i)} + \boldsymbol{z}^{(i)} \right\|_2$. ▷ Corresponds to Eq. 4
    Update parameters: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$.
Output: Trained model $f_{\boldsymbol{\theta}}$.

Algorithm 2 TCCM Inference (Anomaly Scoring)

Input: Test sample $\boldsymbol{z}_{\text{test}}$, trained model $f_{\boldsymbol{\theta}}$, fixed evaluation time $t_{\text{fixed}} \in (0, 1]$ (default $t_{\text{fixed}} = 1$).
Generate time embedding: $\boldsymbol{e}_{t_{\text{fixed}}} \leftarrow \text{SinusoidalEmbedding}(t_{\text{fixed}})$.
Form augmented input: $\tilde{\boldsymbol{z}}_{\text{test}} \leftarrow [\boldsymbol{z}_{\text{test}}; \boldsymbol{e}_{t_{\text{fixed}}}]$.
Predict contraction vector: $\hat{\boldsymbol{v}}_{\text{test}} \leftarrow f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{z}}_{\text{test}})$.
Compute anomaly score: $S(\boldsymbol{z}_{\text{test}}; t_{\text{fixed}}) \leftarrow \|\hat{\boldsymbol{v}}_{\text{test}} + \boldsymbol{z}_{\text{test}}\|_2$. ▷ Corresponds to Eq. 5
Output: Anomaly score $S(\boldsymbol{z}_{\text{test}}; t_{\text{fixed}})$.

B.6 Unsupervised Epoch Selection Strategy

In the main paper, the architecture of TCCM is fixed as a lightweight MLP (2×256 ReLU) across all datasets. While this choice ensures efficiency and comparability, the number of training epochs is not arbitrarily hardcoded. Instead, we adopt a principled and largely automated protocol for unsupervised hyperparameter selection. For completeness, we provide additional details below.

Epoch Selection Protocol. For each dataset, we first examine the empirical training loss curve to identify a rough convergence threshold. Using this as a lower bound, we define a bounded search space of candidate epochs. Within this space, we apply the unsupervised hyperparameter tuning method introduced by Li et al. (2025b), based on the *Improved Contrast Score Margin (CSM)* criterion. This criterion evaluates the margin between top-k predicted anomalous and normal samples solely from the distribution of model outputs, without requiring any ground-truth labels:

$$T(f) = \frac{\hat{\mu}_O - \hat{\mu}_I}{\sqrt{\hat{\sigma}_O^2 + \hat{\sigma}_I^2}},$$

where $\hat{\mu}_O, \hat{\sigma}_O^2$ denote the mean and variance of anomaly scores for the top-k predicted anomalies, and $\hat{\mu}_I, \hat{\sigma}_I^2$ correspond to the remaining $n - k$ presumed inliers. For each candidate epoch, we compute $T(f)$ and select the configuration maximizing this criterion.

Hardcoding for Simplicity. Although this procedure yields dataset-specific and random-seed-dependent epoch values, we ultimately fix the selected epoch across all seeds of a given dataset for simplicity and reproducibility. We note that using per-seed dynamically tuned epochs can sometimes further improve performance, but we chose not to report this to avoid inflating results and to ensure fair comparison across baselines.

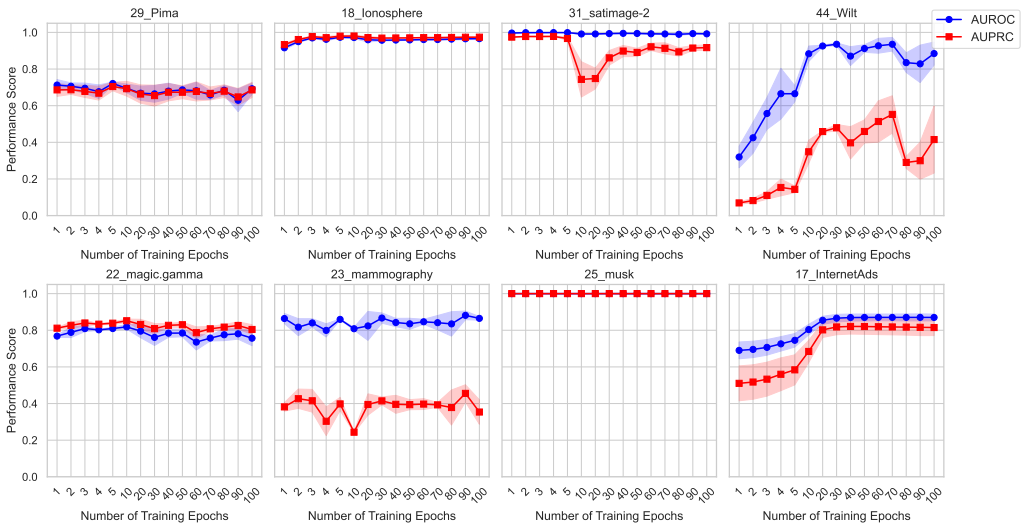


Figure 5: Sensitivity of TCCM to the number of training epochs. For each dataset, we evaluate AUROC and AUPRC across a wide range of epoch values. Results show a **stable plateau** on most datasets (e.g., `Pima`, `Ionosphere`, `Musk`, `InternetAds`), where performance converges early and further training offers minimal gain but adds runtime cost. A similar plateau also appears on `magic.gamma` and `mammography`, albeit with minor fluctuations. On some datasets (e.g., `satimage`), excessive training leads to **overfitting**, while others (e.g., `Wilt`) exhibit stronger fluctuations, reflecting less stable convergence. These findings highlight that: for most datasets, TCCM does not rely on finely tuned epoch numbers and remains robust once a reasonable training horizon is reached.

Sensitivity Analysis. To further address this point, we include a sensitivity analysis over representative datasets. Two consistent patterns emerge:

- **Stable Plateau:** For most datasets (e.g., `Pima`, `Ionosphere`, `Musk`, `InternetAds`), model performance stabilizes after a certain number of epochs, where further training brings little to no improvement but increases runtime. A similar plateau is also observed on `magic.gamma` and `mammography`, though with minor fluctuations.
- **Overfitting Risk:** On some datasets (notably `satimage`), training beyond the plateau results in performance degradation, suggesting overfitting. In contrast, `Wilt` shows larger fluctuations, indicating less stable convergence rather than clear overfitting.

These findings justify our principled choice of using CSM-based epoch selection combined with early convergence boundaries.

Discussion. We emphasize that unsupervised hyperparameter tuning remains an under-explored but important challenge in anomaly detection. Our approach leverages a recently proposed and validated criterion, but we believe future work should explore more adaptive and automated tuning protocols.

C Property Analysis

Philosophical Analogy. Our method encodes a natural inductive bias: no matter where a sample lies along the temporal axis, it is always guided by the same high-level goal—movement toward the origin. This echoes the classical proverb *"Only by staying true to our original aspiration can we reach our final destination"*, embodying a consistency that is both geometrically meaningful and empirically effective. In this section, we provide some theoretical analyses of our anomaly detection framework in addition to the analyses given in the main paper as well as their proof (when applicable).

C.1 Relation to Flow Matching and Diffusion Modeling

Our proposed TCCM can be viewed as a task-specific simplification of flow-based learning frameworks, adapted to the semi-supervised anomaly detection setting.

Conventional flow matching methods (Lipman et al., 2022; Liu et al., 2022) aim to learn a continuous-time vector field $\boldsymbol{v}(\boldsymbol{x}, t)$ that transforms samples from a source to a target distribution, often supervised using interpolated trajectories. Extensions to stochastic generative modeling, such as diffusion models, further describe the evolution of data via a stochastic differential equation (SDE):

$$\frac{d\boldsymbol{x}(t)}{dt} = -\alpha(t)\boldsymbol{x}(t) + \sigma(t)\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{6}$$

where $\alpha(t)$ defines the contraction rate toward the origin, and $\sigma(t)$ controls the injection of Gaussian noise over time. While such stochasticity improves sample diversity in generative tasks, it may hinder anomaly detection—especially in the semi-supervised setting where only normal data is observed. The added noise may obscure the underlying structure of normal instances, reducing their separability from anomalies.

TCCM can be interpreted as a deterministic limit of this framework, where we fix $\alpha(t) := 1$ and set $\sigma(t) := 0$. Moreover, instead of simulating time-evolving trajectories, we directly supervise the velocity field at the initial state $\boldsymbol{x}$, using the fixed target vector $-\boldsymbol{x}$ at each sampled time step. This design retains the inductive bias of contraction toward normality while significantly simplifying training and inference, avoiding both trajectory supervision and numerical integration.

Ablation on Noise Injection during Training. To evaluate the effect of noise, we compare TCCM with a noisy variant that injects Gaussian perturbations during training (emulating the SDE in Eq. 6). As shown in Appendix D.3, noise consistently harms anomaly detection performance across AUPRC and AUROC. This supports our hypothesis that deterministic dynamics better preserve structural regularities in normal data.

C.2 Anomaly Score Expectation under Distributional Shift

To theoretically justify the discriminative behavior of our anomaly score, we analyze its expected value under a Gaussian distributional shift. For notation simplicity, we utilise $f_\theta(\boldsymbol{x}, 1)$ for $f_\theta([\boldsymbol{x}; \text{Embed}(1)])$ and $f_\theta(\boldsymbol{z}, 1)$ for $f_\theta([\boldsymbol{z}; \text{Embed}(1)])$ in the following. We consider the case where the learned contraction field satisfies, for all $\boldsymbol{x} \in \mathbb{R}^d$ in the training set,

$$f_\theta(\boldsymbol{x}, 1) = -\boldsymbol{x} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d). \tag{7}$$

The anomaly score then becomes

$$S(\boldsymbol{x}) = \|f_\theta(\boldsymbol{x}, 1) + \boldsymbol{x}\|_2 = \|\boldsymbol{\epsilon}\|_2. \tag{8}$$

Proposition 3 (Discriminative Power under Gaussian-to-Gaussian Shift)**.** *Let normal samples be drawn from $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I_d)$, and anomalous samples from $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 I_d)$ with $\boldsymbol{\mu} \neq \mathbf{0}$. Assume the learned contraction field satisfies $f_\theta(\boldsymbol{x}, 1) = -\boldsymbol{x} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d)$. Assume that the learned velocity field is mismatched for anomalies. Then the corresponding anomaly scores satisfy:*

$$S(\boldsymbol{x}) \sim \chi_d \cdot \sigma_f, \tag{9}$$

$$S(\boldsymbol{z}) \sim \chi_d(\lambda), \quad \textit{with} \quad \lambda = \frac{\|\boldsymbol{\mu}\|_2^2}{\sigma_f^2}, \tag{10}$$

where χ_d and $\chi_d(\lambda)$ denote the central and non-central chi distributions with d degrees of freedom and non-centrality parameter λ, respectively. Moreover, the expected values satisfy:

$$\mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}, \quad \mathbb{E}[S(\boldsymbol{z})] > \mathbb{E}[S(\boldsymbol{x})]. \tag{11}$$

Proof. For normal samples $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I_d)$, the anomaly score reduces to

$$S(\boldsymbol{x}) = \|f_\theta(\boldsymbol{x}, 1) + \boldsymbol{x}\|_2 = \|\boldsymbol{\epsilon}\|_2,$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d)$. Thus,

$$S(\boldsymbol{x}) \sim \sigma_f \cdot \chi_d,$$

and the expected value is

$$\mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \mathbb{E}[\chi_d] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}.$$

For anomalous samples $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 I_d)$, we again have

$$f_\theta(\boldsymbol{z}, 1) = -\boldsymbol{z} + \boldsymbol{\epsilon}, \quad \Rightarrow \quad S(\boldsymbol{z}) = \|f_\theta(\boldsymbol{z}, 1) + \boldsymbol{z}\|_2 = \|\boldsymbol{\epsilon}\|_2,$$

but now $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d)$ is independent of $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 I_d)$, and $\boldsymbol{\epsilon}$ is added to the fixed vector $\boldsymbol{z}$.

Thus, conditional on a sample $\boldsymbol{z}$, the score becomes

$$S(\boldsymbol{z}) = \|\boldsymbol{z} - \boldsymbol{z} + \boldsymbol{\epsilon}\|_2 = \|\boldsymbol{\epsilon}\|_2,$$

but this is misleading. In practice, the model is trained to approximate contraction on normal data. For anomalous inputs, the field is mismatched, and we express this by assuming:

$$f_\theta(\boldsymbol{z}, 1) = -\boldsymbol{z}_{\text{proj}} + \boldsymbol{\epsilon},$$

with $\boldsymbol{z}_{\text{proj}}$ being the projection of $\boldsymbol{z}$ onto the normal data manifold. Then:

$$S(\boldsymbol{z}) = \|f_\theta(\boldsymbol{z}, 1) + \boldsymbol{z}\|_2 = \|\boldsymbol{z} - \boldsymbol{z}_{\text{proj}} + \boldsymbol{\epsilon}\|_2 .$$

Let $\boldsymbol{\delta} := \boldsymbol{z} - \boldsymbol{z}_{\text{proj}}$, then:

$$S(\boldsymbol{z}) = \|\boldsymbol{\delta} + \boldsymbol{\epsilon}\|_2 .$$

Since $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d)$, and $\boldsymbol{\delta} \in \mathbb{R}^d$ is fixed conditional on $\boldsymbol{z}$, it follows that:

$$S(\boldsymbol{z}) \sim \chi_d(\lambda), \quad \text{with } \lambda = \frac{\|\boldsymbol{\delta}\|_2^2}{\sigma_f^2}.$$

From standard properties of the non-central chi distribution, we know:

$$\mathbb{E}[\chi_d(\lambda)] > \mathbb{E}[\chi_d] \quad \text{for all } \lambda > 0.$$

Thus:

$$\mathbb{E}[S(\boldsymbol{z})] > \mathbb{E}[S(\boldsymbol{x})],$$

completing the proof. □

Proposition 4 (Discriminative Power under GMM-to-Gaussian Shift)**.** *Let normal data be sampled from a Gaussian mixture model (GMM)*

$$\boldsymbol{x} \sim \sum_{r=1}^{R} \pi_r \cdot \mathcal{N}(\boldsymbol{\mu}_r, \sigma^2 I_d), \quad \sum_{r=1}^{R} \pi_r = 1,$$

and assume the learned contraction field satisfies

$$f_\theta(\boldsymbol{x}, 1) = -\boldsymbol{x} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d).$$

Assume that the learned velocity field is mismatched for anomalies. Define the anomaly score as

$$S(\boldsymbol{x}) := \|f_\theta(\boldsymbol{x}, 1) + \boldsymbol{x}\|_2 = \|\boldsymbol{\epsilon}\|_2.$$

Then the anomaly score for normal data follows a central chi distribution:

$$S(\boldsymbol{x}) \sim \chi_d \cdot \sigma_f,$$

and its expected value is

$$\mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}.$$

Let anomalous samples be drawn from $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}_z, \sigma^2 I_d)$, with $\boldsymbol{\mu}_z \notin \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_R\}$. Then the anomaly score for $\boldsymbol{z}$ satisfies

$$S(\boldsymbol{z}) := \|f_\theta(\boldsymbol{z}, 1) + \boldsymbol{z}\|_2 = \|\boldsymbol{\delta} + \boldsymbol{\epsilon}\|_2,$$

where $\boldsymbol{\delta} := \boldsymbol{z} - \boldsymbol{z}_{proj}$ is the mismatch between $\boldsymbol{z}$ and the normal manifold. Then

$$S(\boldsymbol{z}) \sim \chi_d(\lambda), \quad \lambda = \frac{\|\boldsymbol{\delta}\|_2^2}{\sigma_f^2},$$

and the expected anomaly score satisfies

$$\mathbb{E}[S(\boldsymbol{z})] > \mathbb{E}[S(\boldsymbol{x})].$$

Proof. For normal samples $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_r, \sigma^2 I_d)$, the anomaly score is

$$S(\boldsymbol{x}) = \|f_\theta(\boldsymbol{x}, 1) + \boldsymbol{x}\|_2 = \| - \boldsymbol{x} + \boldsymbol{\epsilon} + \boldsymbol{x}\|_2 = \|\boldsymbol{\epsilon}\|_2.$$

Since $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_f^2 I_d)$, it follows that

$$S(\boldsymbol{x}) \sim \sigma_f \cdot \chi_d.$$

Hence,

$$\mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \mathbb{E}[\chi_d] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}.$$

Now consider an anomalous input $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}_z, \sigma^2 I_d)$, not seen during training. The contraction field, trained only on normal GMM components, is not well aligned with $\boldsymbol{z}$. Let $\boldsymbol{z}_{\text{proj}} \in \text{supp}(p_{\text{data}})$ be the closest point on the normal manifold, then we model the output as:

$$f_\theta(\boldsymbol{z}, 1) \approx -\boldsymbol{z}_{\text{proj}} + \boldsymbol{\epsilon} \quad \Rightarrow \quad S(\boldsymbol{z}) = \|\boldsymbol{z} + f_\theta(\boldsymbol{z}, 1)\|_2 = \|\boldsymbol{z} - \boldsymbol{z}_{\text{proj}} + \boldsymbol{\epsilon}\|_2.$$

Let $\boldsymbol{\delta} := \boldsymbol{z} - \boldsymbol{z}_{\text{proj}}$, which is fixed conditioned on $\boldsymbol{z}$, then

$$S(\boldsymbol{z}) \sim \chi_d(\lambda), \quad \text{with } \lambda = \frac{\|\boldsymbol{\delta}\|_2^2}{\sigma_f^2}.$$

It is a known result that for all $\lambda > 0$, the non-central chi distribution satisfies:

$$\mathbb{E}[\chi_d(\lambda)] > \mathbb{E}[\chi_d],$$

implying that

$$\mathbb{E}[S(\boldsymbol{z})] > \mathbb{E}[S(\boldsymbol{x})].$$

□

Proposition 5 (Namely Proposition 2, Discriminative Power under GMM-to-GMM Shift)**.** *Let normal samples be drawn from a Gaussian mixture model:*

$$\boldsymbol{x} \sim \sum_{r=1}^{R} \pi_r \cdot \mathcal{N}(\boldsymbol{\mu}_r, \sigma^2 I_d), \quad \sum_{r=1}^{R} \pi_r = 1.$$

Let anomalous samples be drawn from another Gaussian mixture model with distinct component means:

$$\boldsymbol{z} \sim \sum_{s=1}^{S} \eta_s \cdot \mathcal{N}(\boldsymbol{\nu}_s, \sigma^2 I_d), \quad \sum_{s=1}^{S} \eta_s = 1,$$

with $\boldsymbol{\nu}_s \notin \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_R\}$ for all s. Assume the learned contraction field satisfies:

$$f_\theta(\boldsymbol{x}, 1) = -\boldsymbol{x} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_f^2 I_d).$$

Assume that the learned velocity field is mismatched for anomalies. [10] *Define the anomaly score as:*

$$S(\boldsymbol{x}) := \|f_\theta(\boldsymbol{x}, 1) + \boldsymbol{x}\|_2 .$$

Then:

1. *For normal samples:*

$$S(\boldsymbol{x}) \sim \chi_d \cdot \sigma_f, \quad \mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}.$$

2. *For anomalous samples, each component satisfies:*

$$S(\boldsymbol{z}) \sim \chi_d(\lambda_s), \quad \lambda_s = \frac{\|\boldsymbol{\nu}_s - \boldsymbol{\mu}_{r^*(s)}\|_2^2}{\sigma_f^2},$$

where $\boldsymbol{\mu}_{r^*(s)} := \arg\min_{\boldsymbol{\mu}_r} \|\boldsymbol{\nu}_s - \boldsymbol{\mu}_r\|_2$. *Then:*

$$\mathbb{E}[S(\boldsymbol{z})] = \sum_{s=1}^{S} \eta_s \cdot \mathbb{E}[\chi_d(\lambda_s)] > \mathbb{E}[S(\boldsymbol{x})].$$

Proof. **Step 1: Normal samples.**

For any $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_r, \sigma^2 I_d)$, since the contraction field is learned from normal data, we assume it satisfies:

$$f_\theta(\boldsymbol{x}, 1) = -\boldsymbol{x} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d).$$

Therefore,

$$S(\boldsymbol{x}) = \|f_\theta(\boldsymbol{x}, 1) + \boldsymbol{x}\|_2 = \|\boldsymbol{\epsilon}\|_2 \sim \chi_d \cdot \sigma_f.$$

Thus, for normal samples, the anomaly score distribution is a central chi distribution with scale σ_f. Its expectation is given by:

$$\mathbb{E}[S(\boldsymbol{x})] = \sigma_f \cdot \sqrt{2} \cdot \frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}.$$

Step 2: Anomalous samples.

Each anomalous component is $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\nu}_s, \sigma^2 I_d)$. Since the model is trained only on normal components $\boldsymbol{\mu}_r$, it cannot learn a correct contraction vector for $\boldsymbol{z}$. As an approximation, we model the field as:

$$f_\theta(\boldsymbol{z}, 1) \approx -\boldsymbol{z}_{\text{proj}} + \boldsymbol{\epsilon},$$

where $\boldsymbol{z}_{\text{proj}}$ is the projection of $\boldsymbol{z}$ onto the nearest normal cluster center:

$$\boldsymbol{z}_{\text{proj}} := \boldsymbol{\mu}_{r^*(s)} = \arg\min_{\boldsymbol{\mu}_r} \|\boldsymbol{z} - \boldsymbol{\mu}_r\|_2.$$

Then the anomaly score becomes:

$$S(\boldsymbol{z}) = \|f_\theta(\boldsymbol{z}, 1) + \boldsymbol{z}\|_2 = \|\boldsymbol{z} - \boldsymbol{z}_{\text{proj}} + \boldsymbol{\epsilon}\|_2 .$$

[10] Regarding empirical support for this assumption, we offer three points of clarification: (1) Direct visual validation: Figure 1 provides visualizations of the learned contraction vectors on synthetic 2D datasets. These examples clearly demonstrate that anomalous points consistently deviate from the expected contraction field, validating the mismatch assumption in a controlled and interpretable setting. (2) Indirect support through benchmark results: Across 47 real-world tabular datasets, TCCM consistently achieves strong AUROC and AUPRC scores. This level of performance would be difficult to attain if the model failed to differentiate between normal and anomalous points during inference—thus indirectly supporting the presence and utility of the mismatch behavior assumed in our analysis. (3) Controlled synthetic validation: We aslo provide a dedicated empirical study based on the Gaussian mixture setup. By comparing anomaly score distributions for normal and anomalous points across multiple dimensions ($d = 2, 5, 10, 15, 20$), we show that anomalies consistently yield higher scores. This directly validates the mismatch assumption in a controlled setting aligned with our theoretical analysis.

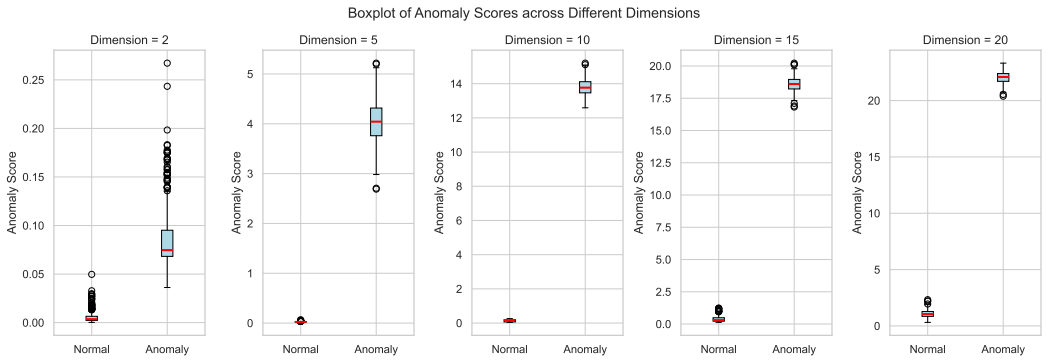


Figure 6: Empirical validation of the mismatch assumption in Propositions 3, 4, and 5. Boxplots show anomaly score distributions for normal and anomalous samples under different data dimensions ($d = 2, 5, 10, 15, 20$). Across all cases, anomalous points consistently yield higher scores than normal points, supporting the assumption that anomalies incur a systematic mismatch under the learned contraction field.

Let $\boldsymbol{\delta}_s := \boldsymbol{z} - \boldsymbol{\mu}_{r^*(s)}$, which satisfies $\boldsymbol{\delta}_s \sim \mathcal{N}(\boldsymbol{\nu}_s - \boldsymbol{\mu}_{r^*(s)}, \sigma^2 I_d)$. Since $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 I_d)$, the sum $\boldsymbol{\delta}_s + \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\nu}_s - \boldsymbol{\mu}_{r^*(s)}, (\sigma^2 + \sigma_f^2) I_d)$. Hence,

$$S(\boldsymbol{z}) \sim \chi_d(\lambda_s), \quad \lambda_s = \frac{\|\boldsymbol{\nu}_s - \boldsymbol{\mu}_{r^*(s)}\|_2^2}{\sigma_f^2}.$$

Then the overall anomaly score distribution for anomalous samples (from the mixture) is:

$$S(\boldsymbol{z}) \sim \sum_{s=1}^{S} \eta_s \cdot \chi_d(\lambda_s), \quad \text{with } \lambda_s > 0.$$

It is a standard result that:

$$\mathbb{E}[\chi_d(\lambda_s)] > \mathbb{E}[\chi_d] \quad \forall \lambda_s > 0.$$

Therefore,

$$\mathbb{E}[S(\boldsymbol{z})] = \sum_{s=1}^{S} \eta_s \cdot \mathbb{E}[\chi_d(\lambda_s)] > \mathbb{E}[\chi_d] = \frac{1}{\sigma_f} \cdot \mathbb{E}[S(\boldsymbol{x})],$$

which implies:

$$\mathbb{E}[S(\boldsymbol{z})] > \mathbb{E}[S(\boldsymbol{x})],$$

completing the proof. □

Empirical Study: Validating the Mismatch Assumption. To empirically validate the key assumption in Propositions 3, 4, and 5—that the learned contraction field is mismatched for anomalies—we conduct an experiment on synthetic Gaussian mixture data. Normal samples are drawn from a GMM with $R = 3$ components (each with isotropic covariance $\sigma^2 I_d$), while anomalous samples are generated from a single Gaussian $\mathcal{N}(\mu_z, \sigma^2 I_d)$ whose mean is located outside the mixture centers. The contraction field f_θ is trained exclusively on normal data using our objective function (see Eq. 4), and anomaly scores $S(x) = \|f_\theta(x, 1) + x\|_2$ are computed for both groups. Figure 6 shows boxplots of the score distributions for normals and anomalies across $d \in \{2, 5, 10, 15, 20\}$. In all cases, anomalous points exhibit consistently higher scores than normal points, with AUROC values exceeding 0.9 regardless of dimension. These results provide direct empirical evidence that anomalies indeed incur a systematic mismatch under the learned contraction field, thereby justifying the modeling assumption made in Propositions 3, 4, and 5.

Limitation of Theoretical Analysis and Future Work. We acknowledge that Proposition 5 is derived under a simplified setting involving Gaussian mixture models (GMMs). We would like to clarify that the use of GMMs in this theoretical result is a deliberate and well-motivated modeling choice:

(1) GMMs have been widely adopted as analytical tools in the machine learning literature—not only in anomaly detection but also in clustering and density modeling. For example, the work of (Zong et al., 2018) introduces the DAGMM model for unsupervised anomaly detection based on similar distributional assumptions. Likewise, GMMs serve as the theoretical backbone in clustering studies such as (Chen and Zhang, 2024), which performs theoretical analysis under anisotropic GMMs. These works demonstrate that GMM-based settings are not only standard but also provide valuable theoretical insight despite being idealized; (2) Our aim is not to suggest that real-world data exactly follow GMMs, but rather to use this setup as a clean, analyzable lens to understand the discriminative behavior of the TCCM scoring function. The result shows that under mild assumptions, the anomaly score is provably larger in expectation for out-of-distribution samples drawn from disjoint mixtures—thus justifying the use of the residual norm as a discriminative signal; (3) Deriving general results under arbitrary data distributions is typically intractable, especially for deep models. Our theoretical analysis strikes a practical balance by providing provable insight under realistic yet analyzable settings. In future work, we aim to explore theoretical extensions to broader classes of distributions, but we believe the current result already provides valuable intuition and justification for the observed empirical behavior.

C.3 Analysis of Representation Collapse

One potential concern for our model is the possibility of *representation collapse*, where the learned mapping trivially reproduces the input or converges to a constant output, thereby failing to distinguish between normal and anomalous data. To verify that TCCM does not suffer from this issue, we provide both theoretical and empirical evidence.

Architectural Considerations. The trivial mapping case, e.g., $f_\theta([\mathbf{z}, \mathrm{Embed}(t)]) = [\mathbf{I}, \mathbf{0}][\mathbf{z}; \mathrm{Embed}(t)]^T = \mathbf{z}$, corresponds to a highly restricted setting where the model degenerates to a single-layer linear transformation without bias or activation, and where the time embedding has no influence. However, this configuration does not reflect the architecture used in TCCM. In practice, TCCM employs a multi-layer MLP with RELU activations and high-dimensional sinusoidal time embeddings that are explicitly concatenated to the input. These design choices allow the model to learn complex, time-varying contraction dynamics, making identity or partial-identity mappings highly unlikely.

Implicit Regularization. Unlike previous methods such as DeepSVDD (Ruff et al., 2018) that prevent collapse by imposing explicit architectural constraints (e.g., bounded activations or bias removal), TCCM avoids such restrictions and instead discourages collapse through *time-conditioned supervision*, multi-time-step optimization, and implicit regularization induced by nonlinear transformations. The temporal embedding ensures that each training instance is contextually distinct across time, which prevents the network from converging to a single trivial representation.

Empirical Verification. To further examine this, we track training dynamics and representation diversity throughout training. Empirically, we observe no evidence of collapse: training loss decreases smoothly without flattening, and anomaly scores exhibit non-degenerate distributions across both normal and anomalous samples. Additionally, the learned feature representations maintain high variance across dimensions, and anomaly detection performance remains stable across datasets (see Figure 2). These observations collectively confirm that TCCM learns meaningful, discriminative representations rather than degenerate identity mappings.

Summary. Overall, TCCM's design—combining multi-layer nonlinear mappings, explicit temporal conditioning, and implicit regularization—effectively mitigates representation collapse without relying on handcrafted architectural constraints. This ensures that the learned contraction field remains expressive and discriminative, supporting robust anomaly detection across diverse data regimes.

D Full Results and Analysis

D.1 Full Analysis of Effectiveness

(1) Effectiveness on Small-scale Dataset. As shown in Table 6, TCCM achieves strong performance in terms of AUROC on small-scale datasets, ranking in the top 10 on 10 out of 12 datasets, with an average rank of 4.42—the best among all evaluated methods. This result highlights the effectiveness of TCCM in low-data regimes. Despite being a deep learning method—which are typically considered data-hungry and prone to underperformance on small datasets—TCCM consistently outperforms both classical and deep baselines. Similar conclusions hold for AUPRC, as shown in Table 7.

(2) Effectiveness on Medium-scale Datasets. As shown in Table 8, TCCM demonstrates strong performance on medium-scale datasets in terms of AUROC, ranking in the top 10 on 13 out of 15 datasets, with an average rank of 6.80—the second best among all anomaly detectors (slightly outperformed by DTE-NonParametric with an average of 6.20). Similarly, Table 9 shows that TCCM ranks in the top 10 on 13 out of 15 datasets in terms of AUPRC, with an average rank of 6.60, achieving the best position overall (followed by DTE-NonParametric with an average rank of 7.13). In both cases, DTE-NonParametric achieves strong performance, but suffers from poor explainability and lacks provable robustness, limiting its practical deployment in sensitive or high-stakes applications.

(3) Effectiveness on Large-scale Datasets. From Table 10, we can see that TCCM gives good performance in terms of AUROC score: it gives top-10 results 9 out of 11 datasets, with an average ranking of 7.36, the second highest ranking among all anomaly detectors (slightly outperformed by DTE-NonParametric with a rank of 6.36). Meanwhile, Table 11 shows that: concerning AUPRC score, it gives top-10 results 9 out of 11 datasets, with an average ranking of 7.18 (followed by DTE-NonParametric with a rank of 8.45), the highest ranking among all anomaly detectors. Note that DTE-NonParametric suffers from low scalability and lack of provable robustness and explainability.

(4) Effectiveness on High-dimensional Datasets. From Table 12, we can see that TCCM gives good performance in terms of AUROC score: it gives top-10 results 9 out of 9 datasets, with an average ranking of 4.89, the second highest ranking among all anomaly detectors (slightly outperformed by DTE-NonParametric with an average rank of 4.44). Meanwhile, Table 13 shows that: concerning AUPRC score, it gives top-10 results on 8 out of 9 datasets, with an average ranking of 5.56 (followed by DTE-NonParametric with a rank of 6.56), the highest ranking among all anomaly detectors. Note that DTE-NonParametric suffers from low scalability at inference and lack of provable robustness and explainability.

D.2 Full Results on Scalability Analysis

D.2.1 Analysis of the Trade-off Between Inference Speed and Accuracy

To further contextualize the efficiency of TCCM, we follow the evaluation practice introduced by DTE (Livernoche et al., 2023) and analyze the relationship between average inference time and detection performance (measured by AUROC and AUPRC) across all 46 anomaly detection methods. As shown in Figures 7 and 8 , TCCM occupies the lower-left region of the plot, indicating simultaneously high accuracy and low inference latency.

Most competing methods fall into one of two regimes: (1) *slow but accurate* models such as DTE-NonParametric, LUNAR, and KDE, which achieve comparable AUROC and AUPRC but require several orders of magnitude longer inference; and (2) *fast but less accurate* methods such as GMM, CBLOF, and Sampling, which exhibit shorter inference but substantially reduced detection accuracy. In contrast, TCCM provides a favorable middle ground—delivering high detection accuracy without compromising inference efficiency.

These results reinforce our claim that TCCM achieves one of the best overall balances between accuracy and computational cost among deep learning-based approaches, demonstrating its strong potential for deployment in real-time and resource-constrained anomaly detection scenarios.

D.2.2 Scalability Analysis on All Algorithms

To assess the practical deployability of TCCM, we perform a comprehensive scalability analysis across three runtime dimensions: **training time**, **inference time**, and **total execution time** (training

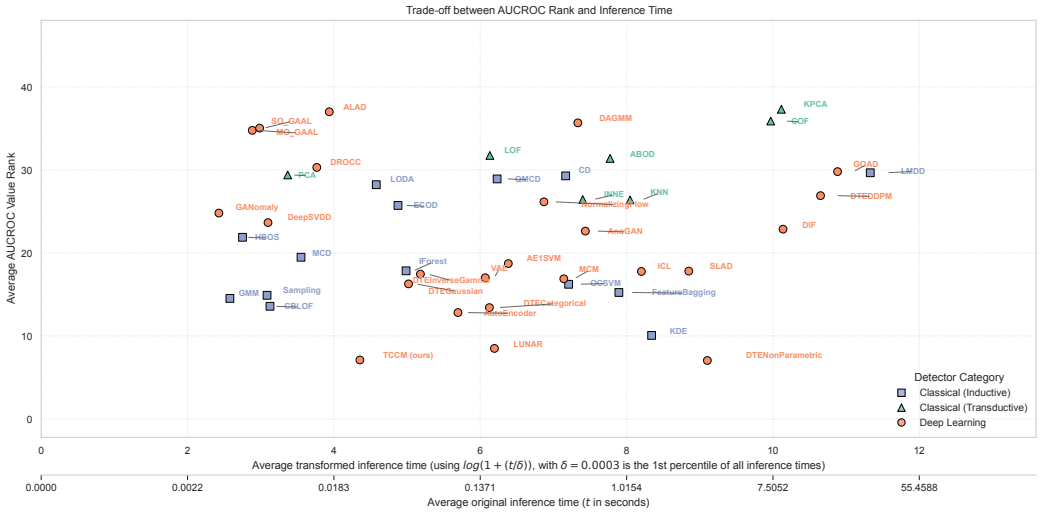


Figure 7: Distribution of *average inference time* (transformed with $\log\left(1 + \frac{t}{\delta}\right)$ to achieve better visualization, with δ is the 1st percentile of all inference times t) vs. *average AUROC rank* across all 45 anomaly detection methods. The ticks corresponding to the original average inference time are also displayed underneath. TCCM achieves the best balance between inference speed and detection accuracy, outperforming both slow but accurate (e.g., DTE-Nonparametric, KDE) and fast but less accurate (e.g., GMM, CBLOF, Sampling) methods.

+ testing). Unlike the main paper, which focuses on comparisons with the strongest deep baselines (DTE-NonParametric, LUNAR, and KDE), here we extend the evaluation to **all 44 baselines**—including classical (transductive and inductive) and deep learning-based methods—to provide a complete view of computational efficiency.

Our analysis centers on **large and high-dimensional datasets**, where runtime differences become most pronounced. Smaller datasets tend to produce negligible timing gaps, as even slower models finish within seconds. In contrast, the large-scale datasets (with hundreds of thousands of samples or high feature dimensionality) amplify differences in efficiency, offering a realistic measure of scalability in deployment settings.

(1) Training Time. As shown in Figure 9, TCCM achieves one of the lowest training times within the deep learning group. Its distribution centers near fast, lightweight models such as AutoEncoder and DeepSVDD, while being faster than most other deep learning methods (e.g., ANOGAN, DTE-Categorical, GOAD). Some classical algorithms (e.g., KDE, OCSVM, LMDD) display higher variability and much longer training durations due to nonparametric or pairwise computations. Overall, TCCM provides a strong balance between model capacity and training efficiency, confirming its practicality for large-scale learning.

(2) Inference Time. Figure 10 presents the distribution of inference times across detectors. Within the deep learning group, TCCM ranks among the fastest methods, close to simple methods such as ALAD and DROCC, but with markedly higher detection accuracy. By contrast, diffusion-based approaches such as DTE-NonParametric, and DTE-DDPM lie at the upper end of the runtime spectrum, often exhibiting multi-order magnitude slower inference across large datasets. Compared to classical baselines (e.g., CBLOF, IFOREST, ECOD), TCCM maintains similar or better inference efficiency while achieving superior detection performance.

(3) Total Runtime. The total runtime (training + inference), summarized in Figure 11, shows that TCCM achieves one of the best overall efficiency–accuracy trade-offs among all evaluated methods. Within the deep learning family, TCCM clusters in the lower range of the runtime distribution, far outperforming heavier diffusion (namely DTE-Gaussian, DTE-InverseGamma) and kernel-based methods (namely KDE). Its compact and stable distribution across large datasets highlights its consistent computational advantage. These results demonstrate that TCCM maintains both scalability and practicality for real-world, high-volume anomaly detection deployments.

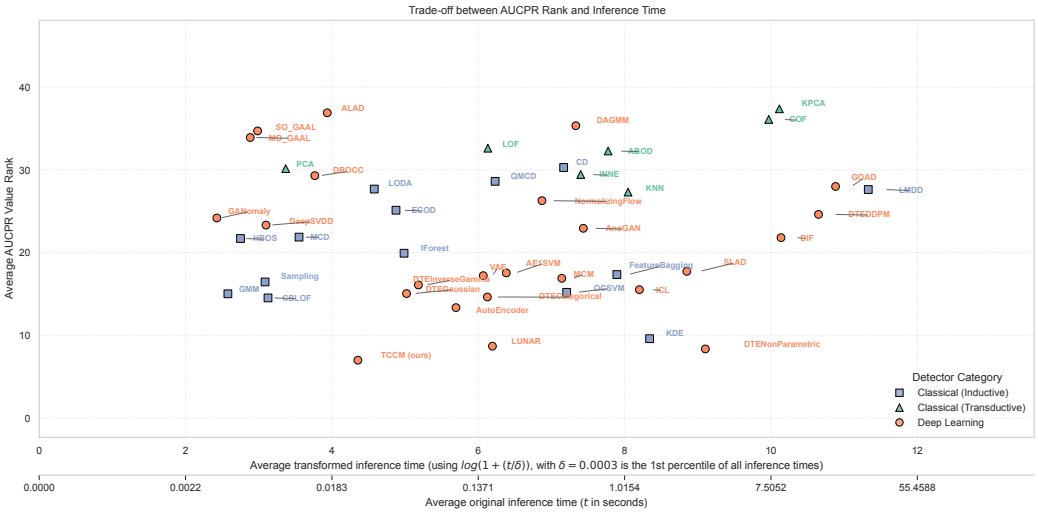


Figure 8: Distribution of *average inference time* (transformed with $\log\left(1+\frac{t}{\delta}\right)$ to achieve better visualization, with δ is the 1st percentile of all all inference times t) vs. *average AUPRC rank* across all 45 anomaly detection methods. The ticks corresponding to the original average inference time are also displayed underneath. TCCM achieves the best balance between inference speed and detection accuracy, outperforming both slow but accurate (e.g., DTE-Nonparametric, KDE) and fast but less accurate (e.g., GMM, CBLOF, Sampling) methods.

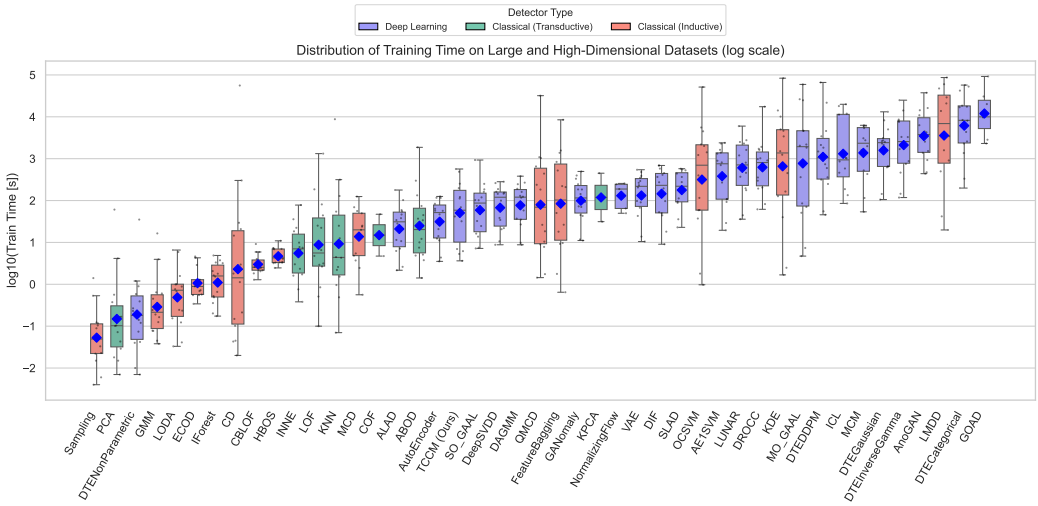


Figure 9: Distribution of training times (log-scale) on 14 large and high-dimensional datasets across all 45 anomaly detectors. TCCM achieves one of the lowest training times among deep learning models, comparable to simple architectures such as AutoEncoder and DeepSVDD, while significantly faster than most other deep learning methods. Some classical models (e.g., KDE, LMDD) show much higher variability and longer training durations.

Discussion on Ultra-Large-Scale Scenarios. While our experiments already include a wide spectrum of realistic datasets—with 14 datasets exceeding 10,000 samples and 6 high-dimensional datasets—two cases are particularly noteworthy: *donors* (619K samples, 10 dimensions) and *census* (299K samples, 500 dimensions). On these datasets, TCCM completes inference in merely **1.05s** and **3.02s**, respectively, whereas the most accurate competitor (DTE-NonParametric) requires **476.60s** and **48,942.85s**. This striking difference (3–4 orders of magnitude) highlights the practical scalability of TCCM in both sample size and feature dimensionality. Furthermore, several conventional and

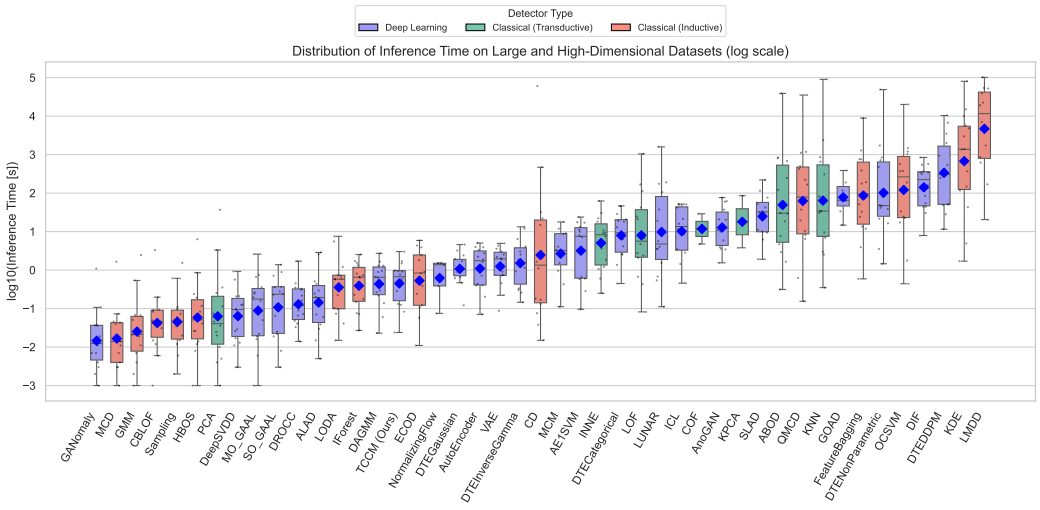


Figure 10: Distribution of inference times (log-scale) on 14 large and high-dimensional datasets. TCCM ranks among the fastest deep learning methods, close to methods such as ALAD and DROCC, but far more accurate. In contrast, diffusion-based baselines (i.e., DTE-NonParametric, DTE-DDPM) occupy the slowest end of the spectrum, illustrating TCCM's superior efficiency for large-scale inference.

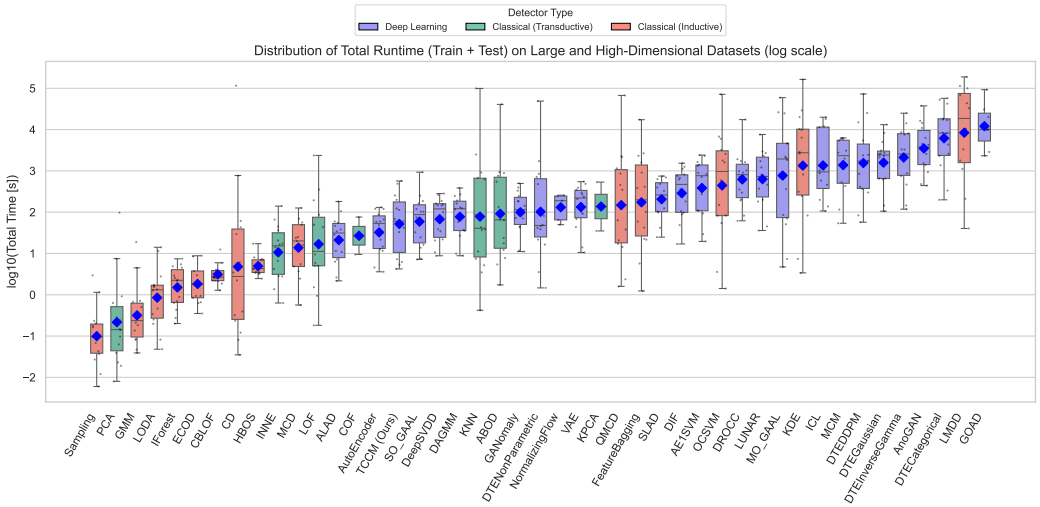


Figure 11: Distribution of total runtimes (training + inference, log-scale) across the 14 large and high-dimensional datasets. TCCM demonstrates one of the best efficiency–accuracy trade-offs within the deep learning category, remaining among the overall fastest detectors. Its compact runtime distribution contrasts sharply with heavier diffusion (namely DTE-Gaussian, DTE-InverseGamma) and kernel-based methods (namely KDE), confirming its scalability and deployability in high-volume environments.

deep baselines fail to process such large-scale inputs within reasonable resource constraints (e.g., memory overflow or exceeding 72h runtime; see Tables 10 and 12). Although evaluating on tens of millions of samples remains an exciting direction for future work, our current results already provide compelling evidence that TCCM is well-suited for real-world, large-scale anomaly detection deployments.

D.3 Ablation Studies and Sensitivity Analysis: Full Results and Analysis

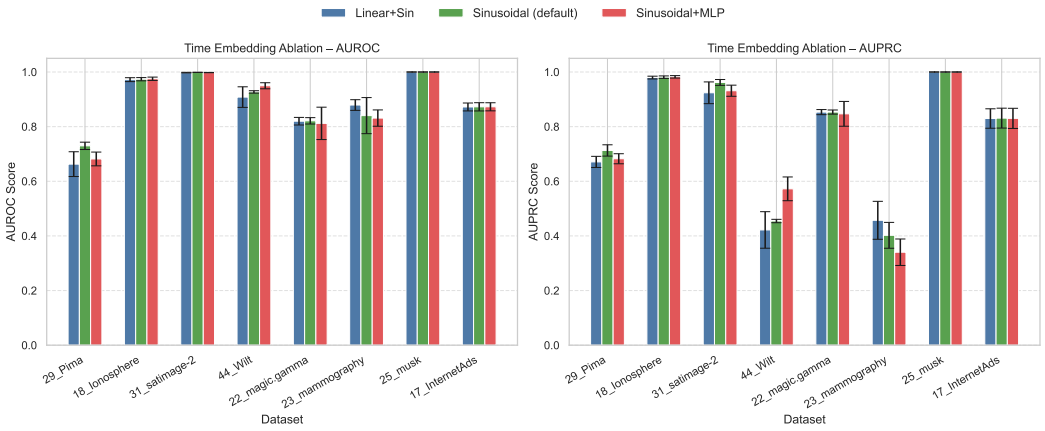


Figure 12: Ablation Study on Time Embedding Methods. We compare three different time embedding strategies used in our flow-based anomaly detection model: *Linear+Sin*, *Sinusoidal (default)*, and *Sinusoidal+MLP*, across eight representative datasets spanning four categories: *Small* (29_Pima, 18_Ionosphere), *Medium* (31_satimage-2, 44_Wilt), *Large* (22_magic.gamma , 23_mammography), and *High-dimensional* (25_musk, 17_InternetAds). The figure shows AUROC (left) and AUPRC (right) scores on the y-axis versus dataset names on the x-axis. Bars are grouped by embedding method and include standard deviation as error bars. Results show that our model is robust across all embedding types, with the default *Sinusoidal* embedding generally offering strong and stable performance.

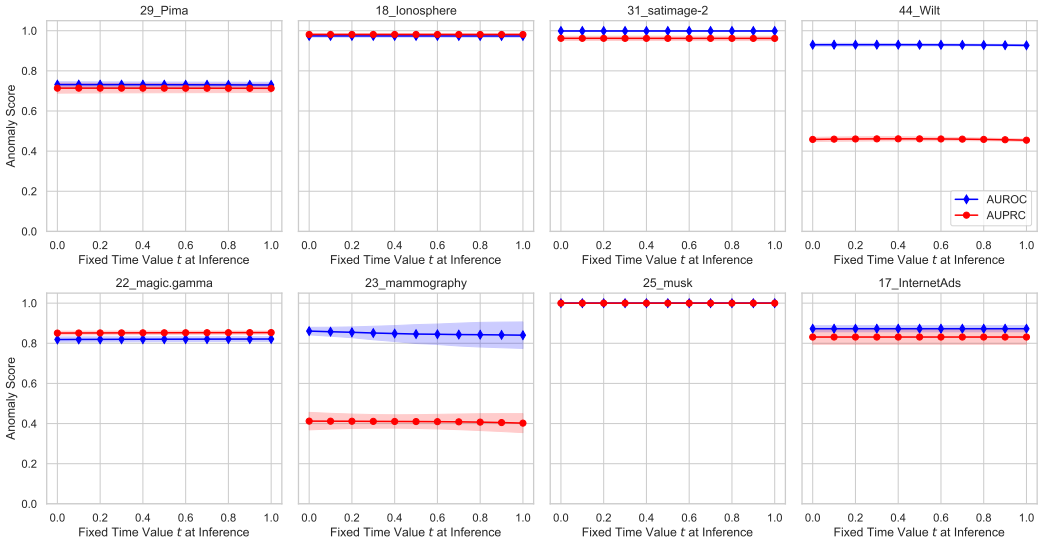


Figure 13: Sensitivity Analysis on Time Value at Inference. We evaluate the sensitivity of our model to different fixed time inputs $t \in [0.0, 1.0]$ at inference across four categories of datasets: *Small* (29_Pima, 18_Ionosphere), *Medium* (31_satimage-2, 44_Wilt), *Large* (22_magic.gamma , 23_mammography), and *High-dimensional* (25_musk, 17_InternetAds). Each plot shows the average AUROC (blue) and AUPRC (red) across 5 random seeds, with individual points marked on each curve. Shaded regions indicate one standard deviation. The **x-axis** represents the fixed value of time t, while the **y-axis** reports the detection performance (AUROC or AUPRC). The results demonstrate that our method is insensitive to the specific choice of t. Other datasets show similar behavior and are omitted for brevity.

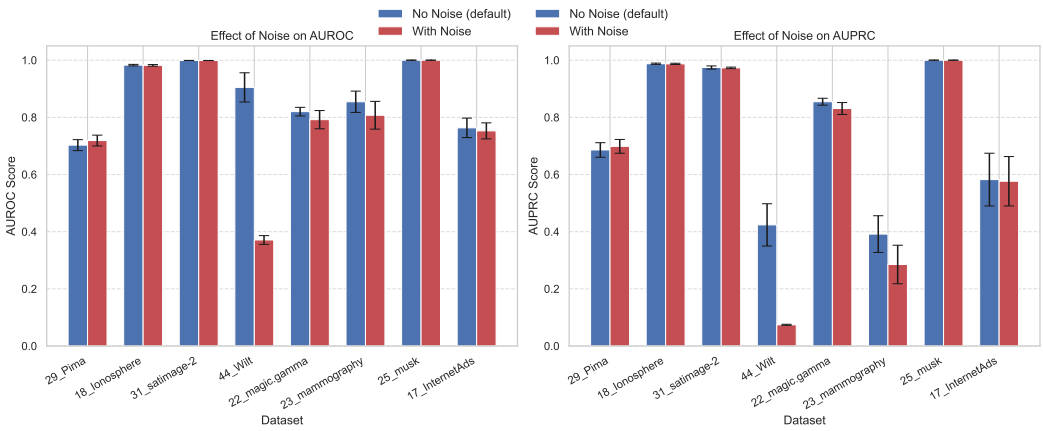


Figure 14: Ablation Study on the Effect of Injecting Noise during Training. We compare the anomaly detection performance (AUROC and AUPRC) of TCCM trained **with** and **without noise** perturbation. We report results across 8 representative datasets spanning four categories (small, medium, large, and high-dimensional). Each bar shows the average score over 5 random seeds, with error bars indicating standard deviation. **Key findings:** (1) On most datasets, adding noise during training does not significantly impact performance; (2) However, in some cases (e.g., `Wilt`, `mammography`), injecting noise leads to a substantial drop in AUROC and/or AUPRC. This indicates that noise injection, while helpful in diffusion based generative modeling, may hinder learning in deterministic anomaly detection tasks.

Study 1: Time Embedding Variants Used in TCCM. We consider three different time embedding strategies within the TCCM architecture, each representing a trade-off between simplicity and expressiveness:

- **Linear + Sin:** This basic approach applies a single linear layer followed by a sine transformation to the scalar time input t. It is defined as $\phi(t) = \sin(Wt + b)$, where W and b are learnable parameters. This encoding is computationally efficient and empirically fast to converge, making it suitable for lightweight applications.
- **Sinusoidal (default):** Inspired by the positional encoding in Transformers, this method maps time to a fixed set of sinusoidal functions at different frequencies. It is defined as

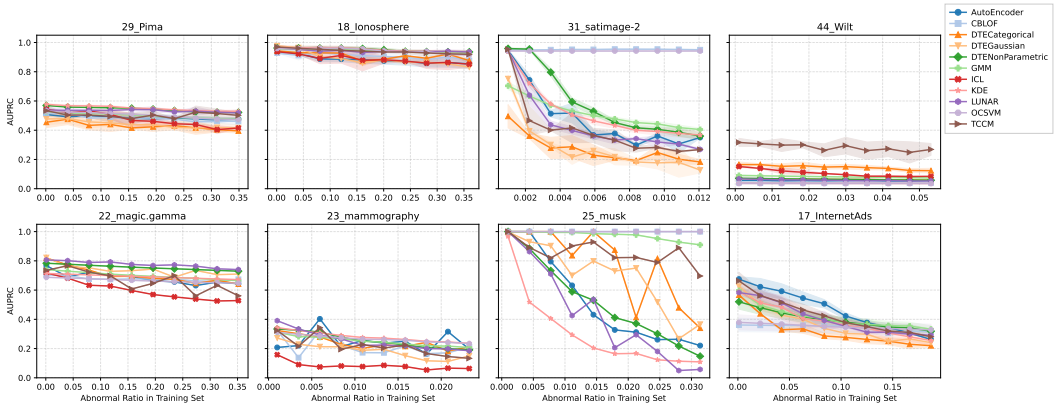
$$\phi(t) = [\sin(\omega_1 t), \cos(\omega_1 t), \ldots, \sin(\omega_d t), \cos(\omega_d t)]\,,$$

  where frequencies ω_i are logarithmically spaced. This embedding captures richer periodic structure without additional learnable parameters.
- **Sinusoidal + MLP:** To enhance the expressiveness of the sinusoidal embedding, we append a two-layer feedforward MLP to it. This allows the model to learn nonlinear combinations of the sinusoidal basis, which is often beneficial when modeling more complex dynamics. However, it introduces more parameters and increases training time.

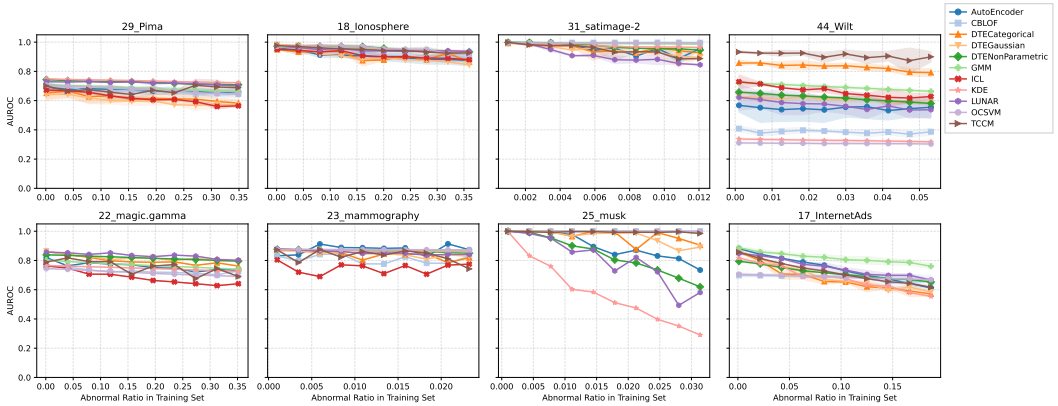
All three variants are seamlessly plugged into the same backbone network, differing only in the time embedding module. In our experiments, we observe consistent performance across them, while the sinusoidal embedding offers a good balance between performance and simplicity.

Study 2: Sensitivity to Fixed Time t during Inference. In the TCCM framework, the final anomaly score is computed based on the model output at a specific time t, typically fixed to $t = 1.0$ during inference. To assess the robustness of our method to the choice of t, we conduct a sensitivity analysis by varying t uniformly in the range $[0.0, 1.0]$ and measuring the performance in terms of AUROC and AUPRC on eight representative datasets.

For each dataset, we fix t to different values and compute the anomaly score as $S(\boldsymbol{x}) = \|f_{\boldsymbol{\theta}}(\boldsymbol{x}, t) + \boldsymbol{x}\|_2$, where $f_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$ is the predicted vector field. This formulation relies on the observation that, for normal samples, the model learns to approximate $f_{\boldsymbol{\theta}}(\boldsymbol{x}, t) \approx -\boldsymbol{x}$, such that the residual becomes small. Anomalous samples, being out-of-distribution, typically incur larger residuals.



(a) AUPRC vs. contamination ratio.



(b) AUROC vs. contamination ratio.

Figure 15: Sensitivity to training-set contamination for TCCM and 10 top-performing baselines. For each dataset, we fix the train/test split by using 50% of normal samples for training and progressively inject anomalies into the training set (up to the dataset's natural anomaly ratio). The x-axis denotes the abnormal ratio in training; curves summarize mean and variance across multiple seeds (see legends for methods). Overall, increasing contamination tends to reduce performance for most methods; TCCM remains among the most robust, though degradation can still occur on some datasets. We present AUPRC (top) and AUROC (bottom) separately to improve readability.

The results (see Figure 13) demonstrate that the detection performance is largely invariant to the specific value of t, indicating that our method is not sensitive to this hyperparameter. This makes the approach more robust and practical, as it avoids the need for tuning t at inference. The shaded areas in the figure denote standard deviation over five random seeds, further confirming the stability of the results.

Study 3: Effect of Noise Injection during Training. To assess the role of stochasticity in our framework, we compare two variants of TCCM: one trained with Gaussian noise injection—motivated by the SDE formulation in Appendix C.1—and another trained deterministically without noise. In the noisy case, input samples are perturbed as $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x} + t\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, but the model is still supervised to predict the residual vector $-\boldsymbol{x}$. This setup preserves the inductive bias toward contraction while introducing input-level stochasticity during training. In contrast, the deterministic variant trains on unperturbed inputs using the same supervision.

Empirical results, summarized in Figure 14, show that noise injection does not consistently improve performance and in many cases leads to a significant drop in AUROC and AUPRC—particularly for datasets such as `Wilt` and `mammography`. These findings validate our theoretical motivation: while noise injection can enhance sample diversity in generative modeling, anomaly detection—especially

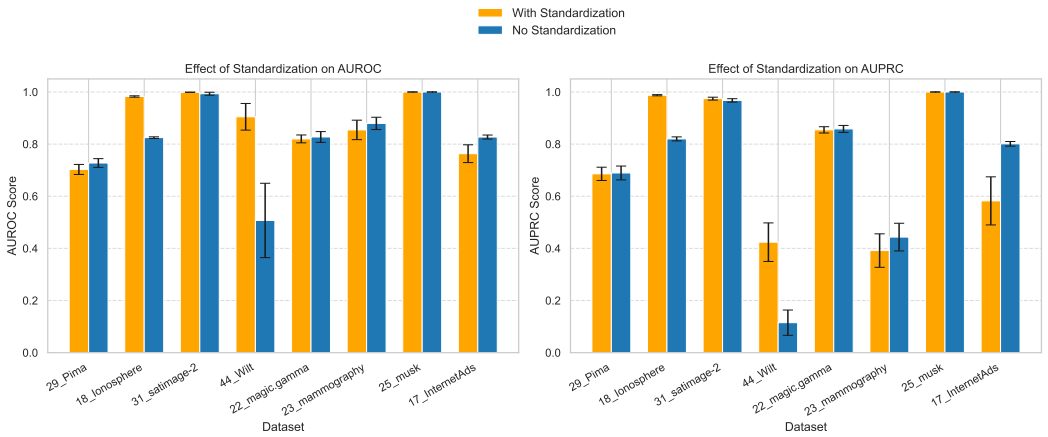


Figure 16: Ablation Study on the Effect of Performing z-score Normalization before Model Training. We compare the anomaly detection performance (AUROC and AUPRC) of TCCM trained **with** and **without noise** z-score normalization. We report results across 8 representative datasets spanning four categories (small, medium, large, and high-dimensional). Each bar shows the average score over 5 random seeds, with error bars indicating standard deviation. **Key findings:** (1) On most datasets, performing z-score normalization does not significantly impact performance; (2) In some cases (e.g., `Ionosphere`, `Wilt`), performing z-score normalization leads to a substantial increase in AUROC and/or AUPRC. This indicates that without normalization, the anomaly scores may be dominated by features with larger scales, leading to suboptimal ranking behavior. However, we note that performing z-score normalization on a few datasets (e.g., `InternetAds`) leads to a drop in AUPRC.

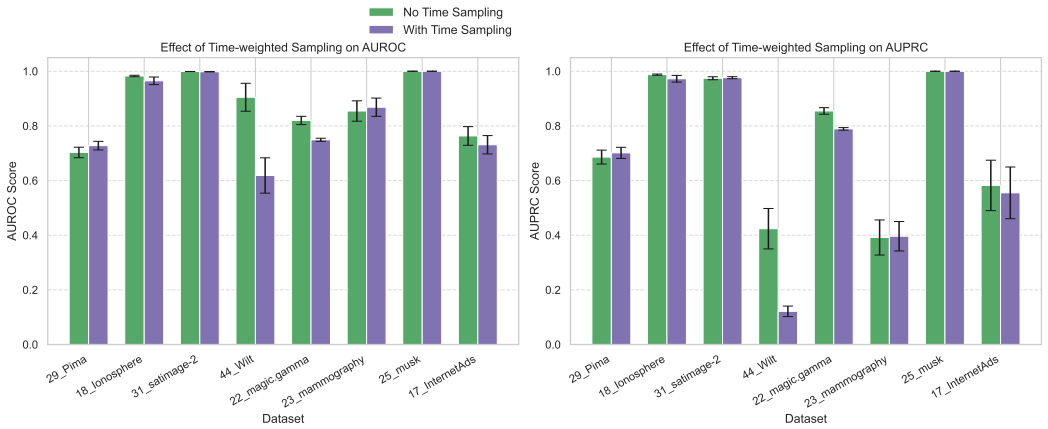


Figure 17: Ablation Study on the Effect of Training with Interpolated Time-Dependent Inputs. We compare the anomaly detection performance (AUROC and AUPRC) of TCCM trained **with** and **without** interpolated inputs $\mathbf{z}_t = t\mathbf{z}$, while keeping all other configurations identical. Results are reported across 8 representative datasets spanning four categories (small, medium, large, and high-dimensional). Each bar shows the average performance over 5 random seeds, with error bars indicating standard deviation. **Key findings:** (1) Introducing time-interpolated samples generally does *not improve* anomaly detection performance, with results remaining approximately unchanged or moderately degraded on most datasets; (2) The degradation is more evident on certain datasets (e.g., `Wilt`, `magic_gamma`), suggesting that interpolated trajectories may introduce undesirable temporal supervision signals; (3) These results empirically support our design choice of directly supervising time-conditioned vector fields at fixed input locations, as discussed in Section 3.

under a semi-supervised regime—relies on preserving the precise structure of normal data. Even mild stochastic perturbations may obscure this structure, weakening the learned vector field. Consequently, our deterministic training procedure yields more stable and effective results for anomaly detection.

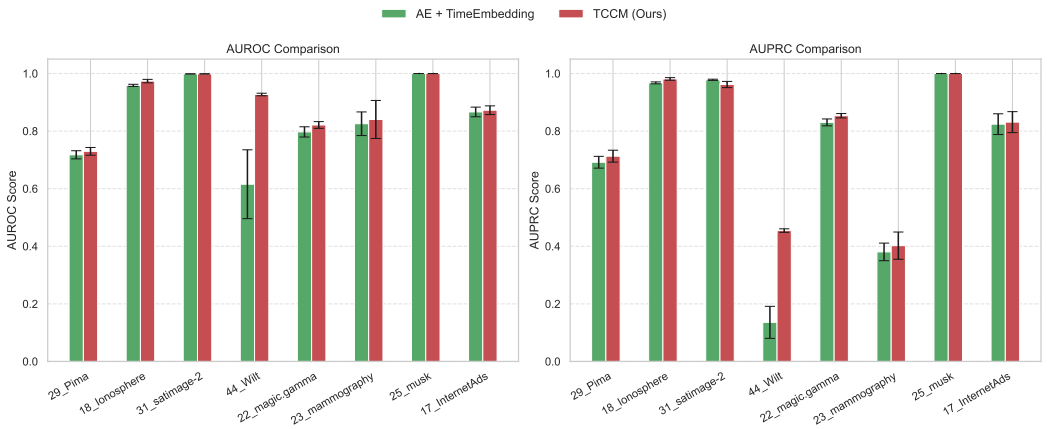


Figure 18: Comparison between TCCM and Autoencoder with Time Embedding (AE+TE) across eight datasets. Bars represent mean AUROC and AUPRC values averaged over 5 random seeds; error bars indicate standard deviation. TCCM consistently performs on par or better, especially on `Wilt` and `magic_gamma`, demonstrating that direct residual learning without reconstruction bottlenecks better captures anomaly-relevant dynamics.

Study 4: Effect of Contamination in Training Data. This study examines how varying degrees of contamination—i.e., abnormal samples present in the training set—affect model performance. Unlike the main experimental setup, where **50% of normal data** is used for training and the test set contains the remaining normal and all abnormal samples, here we **fix the train/test split** by randomly dividing both normal and abnormal data in half: 50% of the normals are used for training, and evaluation is conducted on the remaining normals plus half of the anomalies. We then progressively inject additional anomalies into the training set, increasing the abnormal ratio from near-zero up to each dataset's intrinsic anomaly rate. This protocol normalizes the contamination range across datasets and isolates its effect under the semi-supervised assumption.

Empirical results (Figures 15a–15b) compare TCCM with ten top-performing baselines. While TCCM maintains stable AUROC and AUPRC on several datasets, its performance—like that of most methods—deteriorates as contamination increases, sometimes substantially. This degradation is particularly evident when the abnormal ratio approaches the dataset's natural contamination level, suggesting that even small amounts of anomaly leakage can distort learned decision boundaries. Overall, the results indicate that no method is entirely immune to contaminated supervision and reinforce the practical importance of maintaining a clean training set in semi-supervised anomaly detection.

Study 5: Effectiveness of Conventional Flow Matching on Anomaly Detection. While in principle conventional Flow Matching can be adapted for anomaly detection, our preliminary experiments indicate two natural strategies yield suboptimal results: (i) using a standard Flow Matching model to reconstruct the input $\boldsymbol{x}$ from a learned trajectory and computing the final-step reconstruction error as the anomaly score; (ii) computing a cumulative reconstruction error across multiple time steps along the trajectory. We implemented both approaches and found them to be worse than our proposed TCCM in terms of both detection accuracy and inference efficiency. In particular, trajectory simulation requires numerical integration and multiple model evaluations, incurring high computational cost at inference time. In contrast, TCCM performs anomaly scoring with a single forward pass at a fixed time step, offering both speed and accuracy advantages.

Study 6: Effect of Feature Normalization. In practice, we apply z-score normalization (zero mean, unit variance) to all features before training and inference, which aligns the origin with the center of the normal data distribution. To further evaluate the impact of normalization, we conduct an ablation study comparing TCCM trained **with** and **without** z-score normalization. Results across eight representative datasets (see Figure 16) show that normalization does not significantly affect performance on most datasets but leads to substantial gains in some cases (e.g., `Ionosphere`, `Wilt`). This suggests that without normalization, features with larger scales may dominate the anomaly score,

degrading ranking quality. On a few datasets (e.g., `InternetAds`), normalization slightly reduces AUPRC, indicating dataset-specific effects. Overall, these findings demonstrate that normalization generally improves robustness and provides a principled justification for contracting toward the origin in our framework.

Study 7: Effect of Time-Interpolated Inputs. A key design choice in TCCM is to supervise the time-conditioned vector field directly at fixed input locations, without using time-interpolated samples. To examine whether incorporating interpolated inputs (i.e., $\mathbf{z}_t = t\mathbf{z}$) influences performance, we conduct an ablation study comparing models trained **with** and **without** time interpolation. Results across eight representative datasets (see Figure 17) show that introducing interpolated samples generally does *not improve* anomaly detection performance, with results remaining approximately unchanged or moderately degraded on most datasets. The degradation is more evident on certain datasets (e.g., `Wilt`, `magic_gamma`), suggesting that interpolated trajectories may introduce undesirable or redundant temporal supervision signals, which can interfere with the learning of stable contraction dynamics. Overall, these findings empirically support our design choice of training with fixed inputs and time-conditioned supervision, confirming that TCCM effectively captures temporal dependencies without requiring explicit trajectory interpolation.

Study 8: Distinction Between TCCM and Autoencoder with Time Embedding (AE+TE). A remaining concern pertains to the conceptual distinction between TCCM and an autoencoder (AE) architecture applied to time-augmented data. While both methods may take the same input form $[\mathbf{x}, \text{Embed}(t)]$, their *training objectives*, *architectural principles*, and *learned representations* differ fundamentally.

Conceptual Comparison. Autoencoders aim to minimize a reconstruction loss (e.g., $\|\hat{\mathbf{z}} - \mathbf{z}\|_2^2$), learning to reproduce the input itself. In contrast, TCCM learns a *time-conditioned velocity field* $f_\theta([\mathbf{z}, \text{Embed}(t)])$ that is explicitly supervised toward a fixed contraction direction ($-\mathbf{z}$). This distinction fundamentally alters both the optimization target and the semantics of the learned mapping: TCCM predicts the *instantaneous contraction dynamics* of the data manifold, rather than reconstructing input values. Consequently, the model operates under the framework of *vector field learning*, akin to score-based diffusion or flow-matching methods, not under the reconstruction paradigm of autoencoders.

Architectural Comparison. TCCM predicts the residual vector field directly in the input space using a 3-layer MLP *without* bottleneck compression, maintaining full dimensionality throughout. In contrast, the AE+TE baseline employs a symmetric encoder–decoder architecture with a latent bottleneck layer, formulated as:

- **Encoder:** Linear(input_dim + time_embed_dim $\rightarrow$ 256) $\rightarrow$ ReLU $\rightarrow$ Linear(256 $\rightarrow$ bottleneck_dim) $\rightarrow$ ReLU
- **Decoder:** Linear(bottleneck_dim $\rightarrow$ 256) $\rightarrow$ ReLU $\rightarrow$ Linear(256 $\rightarrow$ input_dim + time_embed_dim)

This bottleneck compression can discard anomaly-related signals, particularly in early training, whereas TCCM preserves feature-level information and learns residual dynamics directly.

Experimental Results. We compare both models on eight representative datasets (see Figure 18), spanning small, medium, large, and high-dimensional settings. TCCM consistently matches or outperforms AE+TimeEmbedding in both AUROC and AUPRC metrics. Notably, the AE+TE baseline exhibits pronounced performance degradation on datasets such as `Wilt` and `magic_gamma`, confirming that its reconstruction-oriented learning objective is less suited for capturing contraction-based anomalies. These results demonstrate that the advantages of TCCM arise not from architectural complexity but from its fundamentally different learning principle.

Conclusion. Both empirically and conceptually, TCCM is *not* an autoencoder. Its flow-inspired supervision, residual prediction mechanism, and non-bottleneck design collectively enable it to model anomaly-relevant dynamics more effectively than reconstruction-based alternatives.

D.4 Empirical Studies on Robustness and Interpretability

D.4.1 Empirical Studies on Robustness

Although has been shown theoretically, we provide an empirical study on robustness here. By following (Bergman and Hoshen, 2020), we utilize PGD (Madry et al., 2017) to create adversarial examples, aiming to make anomalies appear like normal instances (or make normal instances look like anomalies). We measure the increase of false negative rate (i.e., the decrease of anomaly score) or false positive rate (i.e., the increase of anomaly score) on the adversarial examples. To make sure that the attacks are non-trivial, we must limit the allowed budget to use.

Experiment Setup. The experiment is conducted on a suite of synthetic datasets generated from two disjoint Gaussian mixture models. Details of the dataset construction are provided in Table 2. In line with the 65–95–99.7 rule for standard normal distributions, this setup largely satisfies the assumptions of Proposition 5 (namely Proposition 2 in the main paper), while enabling us to evaluate the robustness of the proposed TCCM method. The training set consists of 5,000 samples randomly drawn from the mixture distribution $\mathbb{P}$. The test set comprises 4,000 samples from $\mathbb{P}$ and 1,000 samples from $\mathbb{Q}$, resulting in an anomaly ratio of 0.2. Following our experimental setup used in ADBench, both training and test sets are standardized prior to attack. We consider two types of attacks: 1) False positive attack, where normal samples are perturbed to appear anomalous; 2) False negative attack, where anomalous samples are perturbed to resemble normal data. For the PGD-based attack setup, we evaluate 30 levels of perturbation budgets, $\epsilon \in \{0.1, 0.2, 0.3 \ldots, 2.8, 2.9, 3.0\}$, under the L_∞ norm. Each attack is performed with a step size of 0.01 and a maximum of $\lceil 200 \cdot \epsilon \rceil$ iterations. It is worth noting that, given the data is standardized, a perturbation budget of $\epsilon = 3$ corresponds to a substantial shift in feature space. For both attack types, we track how AUROC and AUPRC evolve with increasing perturbation strength. All experiments are independently repeated five times using different random seeds to ensure statistical robustness.

Table 2: Specifications of synthetic data for robustness verification. I_d is an identity matrix with size d. The normal data is sampled from a GMM with three modes, where $\mu_1 = -3 \times \mathbf{1}_d$, $\mu_2 = \mathbf{0}_d$, and $\mu_3 = 3 \times \mathbf{1}_d$. The anomaly data is sampled from a two-mode GMM, where $\nu_1 = -9 \times \mathbf{1}_d$ and $\nu_1 = 9 \times \mathbf{1}_d$. The experiments are performed across 5 different dimensionalities, i.e., $d \in \{2, 10, 20, 50, 100\}$.

Datasets	Normal $\mathbb{P}$	Anomaly $\mathbb{Q}$
Robustness	$\sum_{r=1}^{3} \frac{1}{3}\mathcal{N}(\mu_r, I_d)$	$\sum_{s=1}^{2} \frac{1}{2}\mathcal{N}(\nu_s, I_d)$

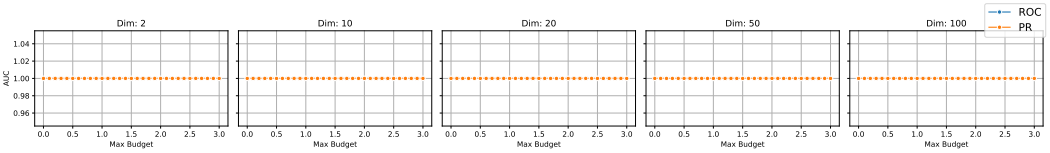


Figure 19: Results of false **negative** attacks (attack on *anomaly* samples) on synthetic GMM-to-GMM shift datasets across five different dimensionalities. The horizontal axis represents the maximum perturbation budget (measured in the L_∞ norm), while the vertical axis indicates the area under the curve (AUC) value.

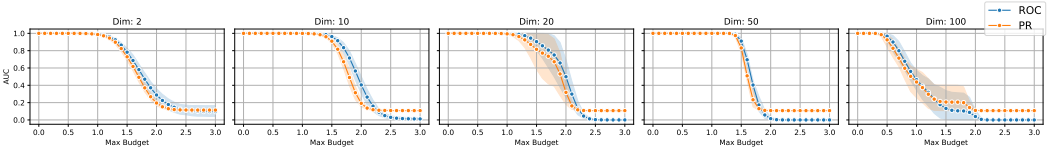


Figure 20: Results of false **positive** attacks (attack on *normal* samples) on synthetic GMM-to-GMM shift datasets across five different dimensionalities. The horizontal axis represents the maximum perturbation budget (measured in the L_∞ norm), while the vertical axis indicates the area under the curve (AUC) value.

The results are presented in Figure 19 and Figure 20, with baseline AUROC and AUPRC scores (i.e., before attack, when the maximum perturbation budget is zero) also indicated for reference. As shown, TCCM is both theoretically justified and empirically validated to be robust under the GMM-to-GMM shift setting. It consistently achieves the highest AUROC and AUPRC across all random seeds and dimensionalities, effectively detecting anomalous samples from $\mathbb{Q}$ in all cases. TCCM demonstrates strong robustness against false negative attacks: both AUROC and AUPRC remain at 1.0, regardless of the perturbation strength. This indicates that adversarial perturbations fail to disguise anomalous inputs as normal. In the case of false positive attacks, TCCM also exhibits a notable degree of robustness. As normal samples are gradually perturbed away from their original distribution, TCCM maintains high AUROC and AUPRC values—particularly up to perturbation levels equivalent to one standard deviation of the standardized data. The only exception occurs in high-dimensional settings (e.g., $d = 100$), where performance slightly degrades. These results suggest that TCCM learns a compact and stable representation of normality, enabling it to ignore semantically meaningless variations within a reasonable margin.

D.4.2 Empirical Studies on Interpretability

To further validate the reliability of TCCM's feature-level importance scores, we conduct a controlled synthetic experiment designed to quantitatively assess whether the learned residual vector field can accurately identify the features responsible for anomalies. This study complements the qualitative analyses in the main paper by providing direct empirical evidence of the model's intrinsic interpretability.

Experimental Design. We construct a well-controlled anomaly detection task based on a Gaussian Mixture Model (GMM) with known anomalous dimensions, enabling precise evaluation of whether TCCM attributes anomalies to the truly perturbed features.

- Normal samples: Drawn from a standard multivariate Gaussian $\mathcal{N}(0, \mathbf{I})$.
- Anomalous samples: Generated from a 3-component GMM:
  - Component 1: 1 dimension shifted,
  - Component 2: 2 dimensions shifted,
  - Component 3: 3 dimensions shifted.
- Shift magnitude: Each shifted feature is perturbed by a random offset uniformly sampled from the range [15, 20].
- Input dimensions: We vary $d \in \{5, 10, 15, 20, 25\}$.
- Training: The model is trained exclusively on normal samples.
- Evaluation: Both anomaly detection and feature-level explanation are assessed on the combined test set.

Evaluation Metrics. We employ two complementary metrics that do not rely on any external explainer:

- **Exact Match:** The proportion of anomalies for which the predicted top-k features *exactly* coincide with the ground-truth anomalous dimensions, where k equals the number of shifted dimensions per sample ($k \in \{1, 2, 3\}$).
- **Jaccard Index:** The average intersection-over-union (IoU) between predicted and true anomalous dimensions across all anomalous samples.

Both metrics are derived directly from the model's built-in residual vector field, computed as $\|[f_\phi([\mathbf{x}, \mathrm{Embed}(t)]) + \mathbf{x}]\|_2$, as defined in Eq. 5 of the main paper. This ensures that interpretability is evaluated based on the model's internal reasoning rather than post hoc approximations.

Results and Discussion. The outcomes in Table 3 show that TCCM consistently and accurately identifies the ground-truth anomalous features across all tested dimensionalities.

The near-perfect ExactMatch and Jaccard scores confirm that TCCM's residual velocity field yields faithful, fine-grained feature-level attributions. Crucially, this interpretability arises *intrinsically* from

Table 3: Quantitative evaluation of explanation accuracy on synthetic GMM anomalies.

Setting	ExactMatch	Jaccard	AUROC	AUPRC
5D	1.000	1.000	1.000	1.000
10D	1.000	1.000	1.000	1.000
15D	1.000	1.000	1.000	1.000
20D	0.996	0.998	1.000	1.000
25D	0.996	0.997	1.000	1.000

the model's formulation—no auxiliary explanation method (e.g., SHAP or LIME) is required. The residual components directly encode each feature's contribution to the contraction mismatch, offering a transparent and actionable view of the decision process. This property enables practitioners in domains such as fraud analysis, medical diagnostics, and industrial monitoring to understand not only *which* samples are anomalous but also *why*.

D.5 Statistical Tests

To rigorously assess whether the performance differences between TCCM and competing methods are statistically significant, we conduct non-parametric statistical tests on their rankings across 47 datasets. For each method, we compute its average AUPRC and AUROC rankings over five random seeds on each dataset. Given the multi-method, multi-dataset nature of this evaluation, traditional pairwise tests are inadequate due to increased risk of Type I error. Hence, we follow the protocol proposed by Demšar (2006), which recommends a two-stage procedure:

- First, we apply the Friedman test (Friedman, 1937), a non-parametric alternative to repeated-measures ANOVA, to determine whether there is any statistically significant difference in performance rankings among all methods.
- If the null hypothesis is rejected, we proceed with the Nemenyi post hoc test (Nemenyi, 1963), which compares all classifiers pairwise. Two methods are considered significantly different if their average ranks differ by at least the critical difference (CD).

Compared to alternatives like the Wilcoxon-Holm method (García et al., 2010), which performs pairwise tests between a control method and others with Holm correction, the Nemenyi test is more conservative—it simultaneously controls the family-wise error rate across all pairwise comparisons, not just against a reference. While this often results in fewer significant findings, it provides a stronger guarantee against false positives, especially important in benchmark settings involving many methods.

We report the results using critical difference diagrams (see Figure 21 and 22). For AUPRC, the Nemenyi test indicates that there are no statistically significant differences among the top-performing group, which includes **TCCM** (ranked 5.8), DTE-NonParametric, LUNAR, KDE, AutoEncoder, ICL, CBLOF, DTE-Categorical, GMM, and OCSVM. For AUROC, the top group includes **TCCM** (ranked 5.7), DTE-NonParametric, LUNAR, KDE, AutoEncoder, ICL, CBLOF, DTE-Categorical, GMM, and Sampling. Although TCCM achieves the best average rank in both metrics, the conservative nature of the Nemenyi test explains the lack of statistically significant superiority. Nonetheless, TCCM consistently ranks at the top, reinforcing its robustness and broad effectiveness across diverse datasets.

Table 4: Overall comparison of top-performant anomaly detection algorithms (with top-4 performance in terms of AUROC and AUPRC) across four key dimensions.

Algorithm	Accuracy	Scalability	Explainability	(Provable) Robustness
TCCM	✓	✓	✓ (feature contribution)	✓
DTE-NonParametric (Livernoche et al., 2023)	✓	✗ (slow inference)	✓(reconstruction)	✗
LUNAR (Goodge et al., 2022)	✓	✗ (slow training)	✗	✗
KDE (Latecki et al., 2007)	✓	✗ (slow training)	✓(density)	✗

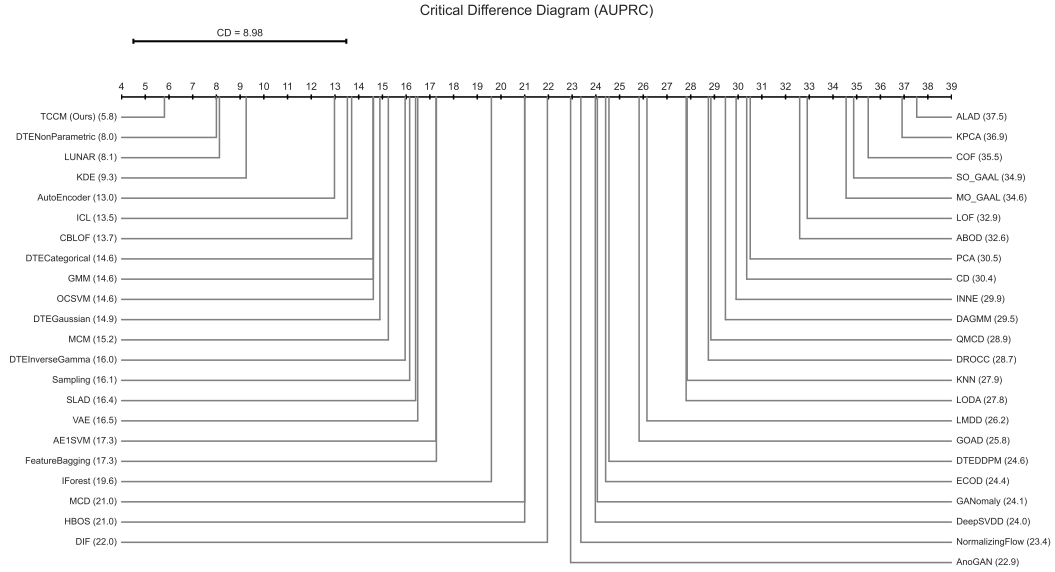


Figure 21: Critical difference (CD) diagram illustrating statistical rank comparisons of the 45 anomaly detection methods based on their AUPRC performance across 47 datasets. Each method is ranked by its mean AUPRC over five random seeds. The CD value, computed via the Nemenyi post-hoc test at significance level 0.05, indicates the minimum difference in average rank that is statistically significant. Notably, TCCM (ranked 5.8 on average) is part of the top-performing group including DTE-NonParametric, LUNAR, KDE, AutoEncoder, ICL, CBLOF, DTE-Categorical, GMM, and OCSVM.

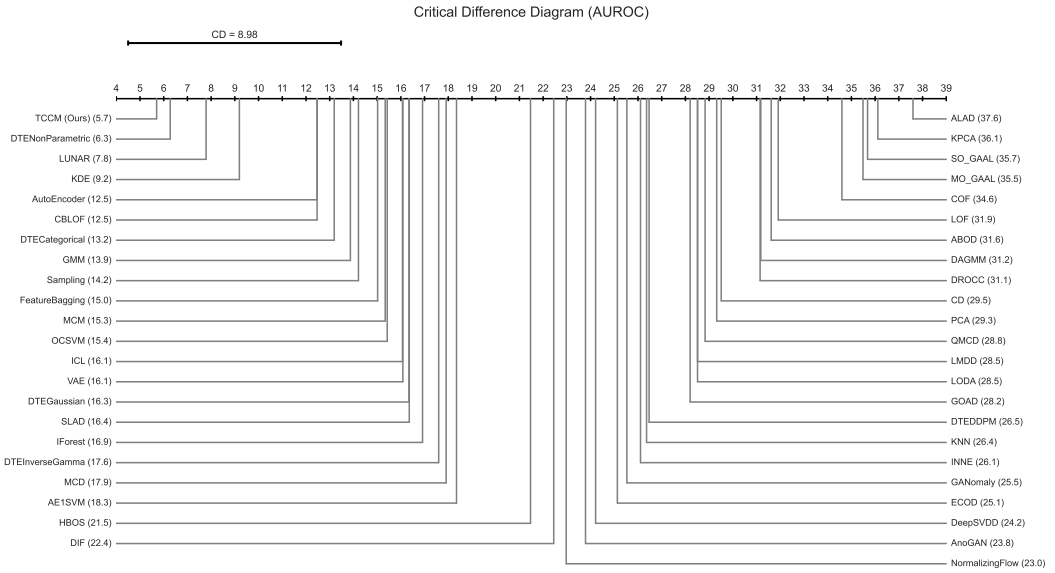


Figure 22: Critical difference (CD) diagram illustrating statistical rank comparisons of the 45 anomaly detection methods based on their AUROC performance across 47 datasets. Each method is ranked by its mean AUROC over five random seeds. The CD value is derived using the Nemenyi post-hoc test with a significance level of 0.05. TCCM (with an average rank of 5.7) belongs to the top-performing group, which includes DTE-NonParametric, LUNAR, KDE, AutoEncoder, ICL, CBLOF, DTE-Categorical, GMM, and Sampling.

D.6 Limitations and Broader Impacts

Limitations. While TCCM achieves state-of-the-art performance with relatively low computational cost, we outline three limitations that offer promising directions for future research. *(1) Data Modality:* As the first work on adapting flow-matching modeling to anomaly detection, our study focuses exclusively on tabular data. Extending TCCM to other data modalities, such as vision (Liu et al., 2024), time series (Blázquez-García et al., 2021), or graph-structured data (Akoglu et al., 2015; Li et al., 2024a), is an exciting avenue for exploration. *(2) Neural Architecture:* To achieve maximum efficiency, TCCM models the velocity field using a multilayer perceptron. This design choice raises an open question: could more sophisticated neural architectures, such as ResNet (He et al., 2016), further improve performance? *(3) Real-World Usability:* Our evaluation is conducted on ADBench (Han et al., 2022), following the common practice in the anomaly detection research community. Exploring TCCM's effectiveness in real-world high-stakes domains, e.g., finance or healthcare, under more dynamic and complex conditions would be valuable.

Broader Impacts. While the TCCM methodology, as presented, is foundational research focused on advancing anomaly detection in tabular data, its limitations inherently shape its potential broader impacts and demarcate avenues for future work that could address these implications. The current focus on tabular data, while demonstrating significant methodological advancements, means that the direct applicability to other prevalent data types like images, time series, or complex graph structures is not yet established. The broader societal impact of anomaly detection often lies in these other domains—such as medical imaging analysis, financial transaction monitoring over time, or social network security. Therefore, until TCCM is extended and validated on these diverse data modalities, its positive impact in such critical areas remains a future prospect, and any potential negative impacts from misuse in these unvalidated contexts are purely speculative but warrant caution.

Furthermore, the utilization of a multilayer perceptron (MLP) for the velocity field, chosen for efficiency, may cap the model's capability to discern highly complex patterns compared to more sophisticated architectures. This architectural limitation could influence its broader impact in scenarios demanding exceptional nuance and accuracy, potentially limiting its deployment in safety-critical applications where the cost of a false negative or positive is extremely high. The ethical implications of deploying a system that might not capture the full complexity of a problem due to architectural constraints should be considered as the research progresses.

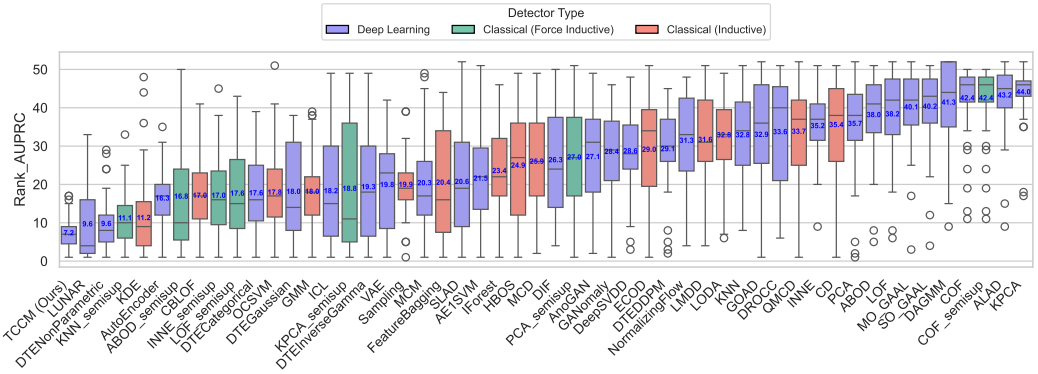
Lastly, the evaluation of TCCM primarily on the ADBench benchmark, though a standard practice, means its performance characteristics in messy, dynamic, and potentially adversarial real-world environments are not fully known. The broader impact, particularly concerning fairness, robustness to unforeseen data shifts, privacy implications in data-sensitive fields like finance or healthcare, and security against emergent threats, can only be truly assessed through rigorous testing in such operational settings. Without this, the translation of TCCM into systems with significant societal touchpoints should proceed with a clear understanding of these unevaluated risks. Future work addressing these limitations will be crucial in responsibly broadening the positive societal impact of this line of research.

E Results under the Inductive Evaluation Setting

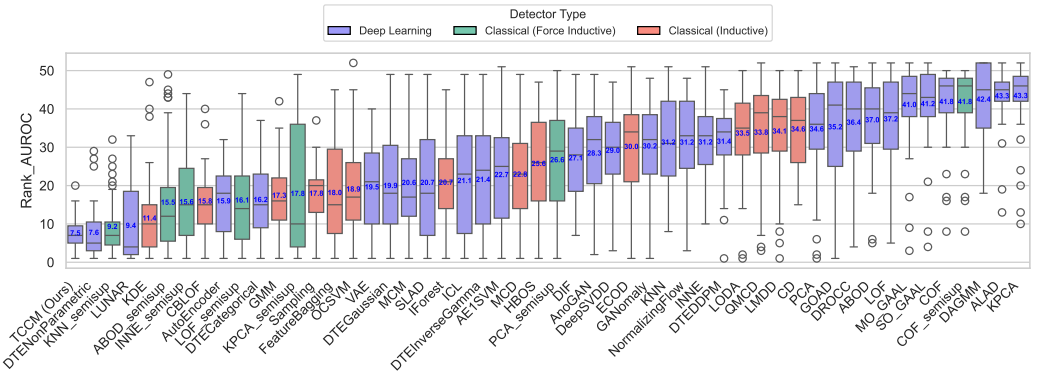
To ensure a protocol that is consistent across *all* methods evaluated together, we additionally report results under a unified inductive (semi-supervised) setting, in which training is performed solely on normal samples without access to anomalous data. Most methods in our benchmark—including the majority of deep learning approaches and many classical baselines—are already inductive by design and thus remain unchanged. For completeness, we *adapt* the following algorithms that were originally formulated as transductive detectors to an inductive training procedure: ABOD, COF, LOF, PCA, KPCA, KNN, and INNE. All other experimental configurations, datasets, and evaluation metrics are identical to those used in the main paper.

E.1 Effectiveness under the Inductive Setting

Figures 23a and 23b summarize the aggregated rankings of 45 detectors across 47 datasets, based on AUPRC and AUROC, respectively. Each ranking averages over five random seeds.



(a) AUPRC ranking distribution across 47 datasets for 45 anomaly detectors under the inductive setting.



(b) AUROC ranking distribution across 47 datasets for 45 anomaly detectors under the inductive setting.

Figure 23: Updated detector ranking distributions (AUPRC and AUROC) under the inductive (semi-supervised) protocol, where models are trained exclusively on normal data. Medians are indicated by horizontal lines; means are shown as numbers.

Findings. The ranking trends for deep learning methods remain highly consistent with the main paper: TCCM continues to rank first overall in both AUPRC and AUROC under the inductive setting, indicating that its performance advantage does not rely on transductive assumptions of other baselines. Other strong baselines (e.g., LUNAR, DTE-NonParametric, KDE) approximately preserve their relative positions.

Notably, several classical methods that we adapted from transductive to inductive exhibit **substantial performance gains**. The improvements are most pronounced for KNN, ABOD, and INNE: for KNN, the average rank in AUPRC improves from 27.9 to 11.1, and in AUROC from 26.4 to 9.2; for ABOD, AUPRC average rank improves from 32.6 to 31.6 and AUROC average rank from 36.1 to 15.5; INNE shows similarly notable gains. These changes indicate that experimental protocol can materially affect certain neighborhood- or structure-based detectors.

E.2 Scalability, Explainability, and Ablation Analyses

Scalability. The scalability conclusions remain aligned with the main paper: TCCM retains its efficiency advantages. Methods that were already inductive (e.g., TCCM, LUNAR, KDE, DTE-NonParametric) are unaffected by the protocol change.

Interpretability. The interpretability analysis in Section 5.1 is unchanged, as TCCM's feature-level explanations stem from its model structure.

Ablation and Sensitivity. We did not repeat ablation/sensitivity studies under the inductive reformulation, since the algorithms modified here (ABOD, COF, LOF, PCA, KPCA, KNN, INNE) were not part of those studies.

Why in the Appendix? We place the inductive-variant results here to keep the main text focused and methodologically consistent: introducing non-canonical inductive variants of originally transductive algorithms into the main tables would complicate the primary comparison without changing our conclusions. The appendix ensures transparency while preserving the clarity of the main results.

In summary, adopting a fully inductive evaluation protocol does not qualitatively change our conclusions. TCCM remains the most effective and scalable detector, with robust performance under semi-supervised training conditions for all baselines.

Table 5: Configuration details for each dataset used in TCCM experiments. To avoid bias from aggressive hyperparameter tuning, we adopt a fixed configuration for all core components (e.g., architecture, time embedding, optimizer) across all datasets. Since our setting is fully unsupervised, we refrain from using label information to optimize hyperparameters. The number of training epochs is adjusted in a dataset-dependent but label-agnostic manner by following the unsupervised internal evaluation strategy in Li et al. (2025b). Note that the backbone architecture is a lightweight 3-layer MLP (two hidden layers), chosen deliberately for efficiency on tabular data; we use the term "deep" in line with common practice to indicate a deep learning-based, end-to-end neural approach rather than architectural depth per se.

Dataset	Category	Architecture	Time Embedding	Loss Function	Optimizer	#Epochs	Batch Size	Seeds
census	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	5	1024	[0,1,2,3,4]
backdoor	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	200	1024	[0,1,2,3,4]
campaign	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	50	1024	[0,1,2,3,4]
mnist	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	500	512	[0,1,2,3,4]
speech	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	500	512	[0,1,2,3,4]
optdigits	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	2000	512	[0,1,2,3,4]
SpamBase	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	5000	512	[0,1,2,3,4]
musk	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	5	512	[0,1,2,3,4]
InternetAds	High-dimensional	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	50	512	[0,1,2,3,4]
donors	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	30	1024	[0,1,2,3,4]
http	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	100	1024	[0,1,2,3,4]
cover	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	10	1024	[0,1,2,3,4]
fraud	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	75	1024	[0,1,2,3,4]
skin	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	110	1024	[0,1,2,3,4]
celeba	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	2	1024	[0,1,2,3,4]
smtp	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	2	1024	[0,1,2,3,4]
ALOI	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	100	1024	[0,1,2,3,4]
shuttle	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	200	1024	[0,1,2,3,4]
magic.gamma	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	10	1024	[0,1,2,3,4]
mammography	Large	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	20	1024	[0,1,2,3,4]
annthyroid	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	2000	512	[0,1,2,3,4]
pendigits	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	1000	512	[0,1,2,3,4]
satellite	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	10	512	[0,1,2,3,4]
landsat	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	6	512	[0,1,2,3,4]
satimage-2	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	5	512	[0,1,2,3,4]
PageBlocks	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	1800	512	[0,1,2,3,4]
Wilt	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	20	512	[0,1,2,3,4]
thyroid	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	10	512	[0,1,2,3,4]
Waveform	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	580	512	[0,1,2,3,4]
Cardiotocography	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	1	512	[0,1,2,3,4]
fault	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	5000	512	[0,1,2,3,4]
cardio	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	2000	512	[0,1,2,3,4]
letter	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	50	512	[0,1,2,3,4]
yeast	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	130	512	[0,1,2,3,4]
vowels	Medium	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	20	512	[0,1,2,3,4]
Pima	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	5	512	[0,1,2,3,4]
breastw	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	1	512	[0,1,2,3,4]
WDBC	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	2	512	[0,1,2,3,4]
Ionosphere	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	10	512	[0,1,2,3,4]
Stamps	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	200	512	[0,1,2,3,4]
vertebral	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	25	512	[0,1,2,3,4]
WBC	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	1	512	[0,1,2,3,4]
glass	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	200	512	[0,1,2,3,4]
WPBC	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	6	512	[0,1,2,3,4]
Lymphography	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	3	512	[0,1,2,3,4]
wine	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	20	512	[0,1,2,3,4]
Hepatitis	Small	MLP (2×256 ReLU)	Sinusoidal (128)	MSE Loss	Adam (lr=0.005)	1	512	[0,1,2,3,4]

Table 6: AUROC results on 12 small datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank).

Dataset	TCMM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
Pima	$0.735_{\pm0.018}$(4)	$0.698_{\pm0.016}$(17)	$0.668_{\pm0.009}$(23)	$0.528_{\pm0.018}$(41)	$0.695_{\pm0.057}$(18)	NAN	$0.686_{\pm0.027}$(19)	$0.604_{\pm0.021}$(32)	$0.429_{\pm0.157}$(42)	$0.651_{\pm0.02}$(27)	$0.569_{\pm0.019}$(37)	$0.66_{\pm0.021}$(25)
breastw	$0.99_{\pm0.003}$(5)	$0.983_{\pm0.006}$(20)	$0.984_{\pm0.004}$(19)	$0.649_{\pm0.02}$(35)	$0.994_{\pm0.001}$(2)	NAN	$0.986_{\pm0.004}$(15)	$0.777_{\pm0.065}$(30)	$0.383_{\pm0.315}$(41)	$0.938_{\pm0.011}$(28)	$0.96_{\pm0.01}$(26)	$0.929_{\pm0.032}$(29)
WDBC	$0.993_{\pm0.003}$(3)	$0.993_{\pm0.005}$(3)	$0.98_{\pm0.023}$(24)	$0.615_{\pm0.06}$(41)	$0.99_{\pm0.006}$(12)	NAN	$0.987_{\pm0.01}$(18)	$0.758_{\pm0.034}$(39)	$0.332_{\pm0.377}$(43)	$0.97_{\pm0.01}$(31)	$0.88_{\pm0.064}$(38)	$0.992_{\pm0.005}$(7)
Ionosphere	$0.976_{\pm0.005}$(3)	$0.948_{\pm0.004}$(16)	$0.934_{\pm0.012}$(19)	$0.542_{\pm0.08}$(44)	$0.778_{\pm0.04}$(34)	NAN	$0.85_{\pm0.042}$(30)	$0.933_{\pm0.007}$(20)	$0.687_{\pm0.167}$(39)	$0.941_{\pm0.011}$(17)	$0.843_{\pm0.029}$(32)	$0.972_{\pm0.004}$(6)
Stamps	$0.935_{\pm0.018}$(8)	$0.926_{\pm0.016}$(14)	$0.944_{\pm0.021}$(4)	$0.597_{\pm0.095}$(41)	$0.741_{\pm0.352}$(35)	NAN	$0.951_{\pm0.021}$(2)	$0.935_{\pm0.02}$(8)	$0.352_{\pm0.151}$(45)	$0.905_{\pm0.026}$(21)	$0.76_{\pm0.052}$(31)	$0.822_{\pm0.109}$(29)
vertebral	$0.669_{\pm0.072}$(1)	$0.457_{\pm0.026}$(22)	$0.53_{\pm0.041}$(9)	$0.48_{\pm0.077}$(18)	$0.566_{\pm0.104}$(6)	NAN	$0.418_{\pm0.058}$(31)	$0.541_{\pm0.035}$(8)	$0.435_{\pm0.265}$(29)	$0.527_{\pm0.072}$(10)	$0.418_{\pm0.043}$(31)	$0.518_{\pm0.019}$(11)
WBC	$0.986_{\pm0.008}$(8)	$0.979_{\pm0.012}$(16)	$0.975_{\pm0.016}$(23)	$0.588_{\pm0.152}$(40)	$0.991_{\pm0.009}$(4)	NAN	$0.98_{\pm0.012}$(15)	$0.762_{\pm0.047}$(38)	$0.377_{\pm0.316}$(42)	$0.92_{\pm0.057}$(32)	$0.906_{\pm0.056}$(33)	$0.962_{\pm0.021}$(26)
glass	$0.903_{\pm0.019}$(2)	$0.748_{\pm0.058}$(27)	$0.763_{\pm0.038}$(22)	$0.501_{\pm0.15}$(44)	$0.694_{\pm0.212}$(32)	NAN	$0.67_{\pm0.125}$(35)	$0.872_{\pm0.016}$(6)	$0.757_{\pm0.173}$(24)	$0.805_{\pm0.048}$(17)	$0.683_{\pm0.1}$(34)	$0.546_{\pm0.032}$(43)
WPBC	$0.567_{\pm0.013}$(2)	$0.516_{\pm0.024}$(20)	$0.495_{\pm0.033}$(34)	$0.488_{\pm0.067}$(37)	$0.535_{\pm0.063}$(16)	NAN	$0.499_{\pm0.063}$(32)	$0.471_{\pm0.029}$(43)	$0.474_{\pm0.071}$(42)	$0.548_{\pm0.038}$(10)	$0.537_{\pm0.054}$(15)	$0.487_{\pm0.018}$(38)
Lymphography	$0.992_{\pm0.008}$(6)	$0.985_{\pm0.019}$(22)	$0.979_{\pm0.028}$(27)	$0.608_{\pm0.166}$(40)	$0.991_{\pm0.008}$(8)	NAN	$0.97_{\pm0.038}$(32)	$0.899_{\pm0.065}$(38)	$0.434_{\pm0.311}$(42)	$0.986_{\pm0.014}$(21)	$0.971_{\pm0.018}$(31)	$0.981_{\pm0.015}$(25)
wine	$0.976_{\pm0.01}$(5)	$0.94_{\pm0.023}$(15)	$0.942_{\pm0.033}$(14)	$0.483_{\pm0.156}$(37)	$0.828_{\pm0.125}$(26)	NAN	$0.825_{\pm0.092}$(27)	$0.665_{\pm0.078}$(33)	$0.427_{\pm0.296}$(40)	$0.957_{\pm0.035}$(13)	$0.629_{\pm0.139}$(35)	$0.896_{\pm0.047}$(24)
Hepatitis	$0.831_{\pm0.031}$(6)	$0.831_{\pm0.027}$(6)	NAN	$0.466_{\pm0.065}$(44)	$0.729_{\pm0.135}$(27)	NAN	$0.764_{\pm0.069}$(23)	$0.707_{\pm0.056}$(30)	$0.477_{\pm0.206}$(43)	$0.81_{\pm0.036}$(15)	$0.695_{\pm0.073}$(32)	$0.776_{\pm0.035}$(20)
Avg Ranking	4.42	16.5	19.82	38.5	18.33	NAN	23.25	27.08	39.33	20.17	31.25	23.58

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
Pima	$0.62_{\pm0.016}$(30)	$0.742_{\pm0.015}$(2)	$0.594_{\pm0.038}$(36)	$0.385_{\pm0.07}$(44)	$0.651_{\pm0.012}$(27)	$0.727_{\pm0.022}$(6)	$0.721_{\pm0.027}$(9)	$0.363_{\pm0.064}$(45)	$0.71_{\pm0.029}$(13)	$0.558_{\pm0.019}$(39)	$0.425_{\pm0.088}$(43)	$0.681_{\pm0.008}$(20)
breastw	$0.689_{\pm0.189}$(33)	$0.987_{\pm0.005}$(14)	$0.951_{\pm0.017}$(27)	$0.721_{\pm0.16}$(32)	$0.965_{\pm0.01}$(25)	$0.989_{\pm0.005}$(8)	$0.986_{\pm0.007}$(15)	$0.04_{\pm0.049}$(45)	$0.971_{\pm0.012}$(24)	$0.99_{\pm0.003}$(5)	$0.23_{\pm0.171}$(44)	$0.99_{\pm0.003}$(5)
WDBC	$0.986_{\pm0.008}$(20)	$0.991_{\pm0.005}$(10)	$0.916_{\pm0.076}$(36)	$0.989_{\pm0.008}$(14)	$0.984_{\pm0.013}$(21)	$0.994_{\pm0.004}$(1)	$0.955_{\pm0.062}$(32)	$0.045_{\pm0.02}$(45)	$0.981_{\pm0.007}$(22)	$0.979_{\pm0.007}$(25)	$0.155_{\pm0.121}$(44)	$0.988_{\pm0.009}$(16)
Ionosphere	$0.965_{\pm0.014}$(7)	$0.978_{\pm0.005}$(1)	$0.908_{\pm0.057}$(26)	$0.932_{\pm0.015}$(21)	$0.962_{\pm0.009}$(10)	$0.976_{\pm0.007}$(3)	$0.676_{\pm0.1}$(42)	$0.686_{\pm0.092}$(40)	$0.941_{\pm0.009}$(17)	$0.957_{\pm0.006}$(14)	$0.703_{\pm0.084}$(38)	$0.91_{\pm0.021}$(25)
Stamps	$0.782_{\pm0.114}$(30)	$0.949_{\pm0.014}$(3)	$0.885_{\pm0.07}$(24)	$0.609_{\pm0.252}$(40)	$0.864_{\pm0.025}$(27)	$0.941_{\pm0.023}$(5)	$0.921_{\pm0.026}$(17)	$0.628_{\pm0.225}$(38)	$0.902_{\pm0.018}$(22)	$0.621_{\pm0.058}$(39)	$0.757_{\pm0.135}$(32)	$0.933_{\pm0.02}$(11)
vertebral	$0.616_{\pm0.05}$(2)	$0.45_{\pm0.017}$(25)	$0.457_{\pm0.085}$(22)	$0.449_{\pm0.071}$(27)	$0.505_{\pm0.044}$(12)	$0.45_{\pm0.035}$(25)	$0.462_{\pm0.122}$(21)	$0.612_{\pm0.108}$(3)	$0.581_{\pm0.105}$(5)	$0.479_{\pm0.034}$(19)	$0.594_{\pm0.118}$(4)	$0.487_{\pm0.024}$(16)
WBC	$0.806_{\pm0.162}$(37)	$0.981_{\pm0.007}$(14)	$0.924_{\pm0.059}$(30)	$0.893_{\pm0.054}$(35)	$0.935_{\pm0.023}$(29)	$0.978_{\pm0.01}$(18)	$0.978_{\pm0.014}$(18)	$0.024_{\pm0.029}$(45)	$0.959_{\pm0.024}$(27)	$0.987_{\pm0.008}$(7)	$0.064_{\pm0.07}$(44)	$0.986_{\pm0.009}$(8)
glass	$0.658_{\pm0.085}$(36)	$0.879_{\pm0.013}$(5)	$0.761_{\pm0.095}$(23)	$0.611_{\pm0.187}$(42)	$0.919_{\pm0.019}$(1)	$0.89_{\pm0.02}$(3)	$0.833_{\pm0.037}$(13)	$0.625_{\pm0.234}$(40)	$0.829_{\pm0.055}$(14)	$0.844_{\pm0.032}$(12)	$0.626_{\pm0.2}$(39)	$0.724_{\pm0.035}$(30)
WPBC	$0.506_{\pm0.036}$(28)	$0.561_{\pm0.029}$(5)	$0.545_{\pm0.037}$(11)	$0.564_{\pm0.035}$(3)	$0.511_{\pm0.049}$(24)	$0.564_{\pm0.028}$(3)	$0.495_{\pm0.063}$(34)	$0.482_{\pm0.031}$(40)	$0.502_{\pm0.037}$(29)	$0.542_{\pm0.011}$(14)	$0.499_{\pm0.049}$(32)	$0.509_{\pm0.024}$(25)
Lymphography	$0.977_{\pm0.013}$(28)	$0.99_{\pm0.009}$(14)	$0.97_{\pm0.016}$(32)	$0.981_{\pm0.023}$(25)	$0.97_{\pm0.025}$(32)	$0.987_{\pm0.005}$(20)	$0.992_{\pm0.006}$(6)	$0.17_{\pm0.263}$(44)	$0.973_{\pm0.038}$(29)	$0.989_{\pm0.011}$(17)	$0.259_{\pm0.324}$(43)	$0.993_{\pm0.008}$(4)
wine	$0.925_{\pm0.03}$(20)	$0.972_{\pm0.015}$(7)	$0.646_{\pm0.282}$(34)	$0.963_{\pm0.015}$(9)	$0.906_{\pm0.026}$(22)	$0.978_{\pm0.016}$(3)	$0.776_{\pm0.242}$(28)	$0.153_{\pm0.149}$(43)	$0.903_{\pm0.07}$(23)	$0.977_{\pm0.011}$(4)	$0.106_{\pm0.113}$(44)	$0.935_{\pm0.023}$(17)
Hepatitis	$0.767_{\pm0.062}$(22)	$0.831_{\pm0.025}$(6)	$0.725_{\pm0.065}$(28)	$0.833_{\pm0.029}$(3)	$0.7_{\pm0.066}$(31)	$0.763_{\pm0.084}$(24)	$0.786_{\pm0.049}$(18)	$0.59_{\pm0.076}$(40)	$0.625_{\pm0.092}$(35)	$0.776_{\pm0.041}$(20)	$0.535_{\pm0.081}$(42)	$0.833_{\pm0.02}$(3)
Avg Ranking	24.42	8.83	27.42	24.58	21.75	9.92	21.08	39.0	21.67	17.92	37.42	15.0

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
Pima	$0.71_{\pm0.018}$(13)	$0.676_{\pm0.011}$(22)	$0.595_{\pm0.016}$(35)	$0.706_{\pm0.016}$(15)	$0.72_{\pm0.015}$(11)	$0.727_{\pm0.017}$(6)	$0.731_{\pm0.012}$(5)	$0.754_{\pm0.017}$(1)	$0.599_{\pm0.079}$(33)	$0.654_{\pm0.076}$(26)	$0.719_{\pm0.014}$(12)	$0.701_{\pm0.017}$(16)
breastw	$0.989_{\pm0.003}$(8)	$0.976_{\pm0.003}$(22)	$0.991_{\pm0.001}$(3)	$0.581_{\pm0.242}$(38)	$0.985_{\pm0.004}$(18)	$0.991_{\pm0.002}$(3)	$0.995_{\pm0.001}$(1)	$0.989_{\pm0.005}$(8)	$0.628_{\pm0.076}$(36)	$0.981_{\pm0.01}$(21)	$0.989_{\pm0.003}$(8)	$0.989_{\pm0.004}$(8)
WDBC	$0.989_{\pm0.004}$(14)	$0.942_{\pm0.008}$(34)	$0.971_{\pm0.007}$(30)	$0.994_{\pm0.003}$(1)	$0.99_{\pm0.004}$(12)	$0.987_{\pm0.004}$(18)	$0.991_{\pm0.004}$(10)	$0.993_{\pm0.004}$(3)	$0.992_{\pm0.007}$(7)	$0.974_{\pm0.017}$(28)	$0.975_{\pm0.005}$(26)	$0.992_{\pm0.006}$(7)
Ionosphere	$0.961_{\pm0.01}$(12)	$0.919_{\pm0.004}$(23)	$0.732_{\pm0.011}$(37)	$0.95_{\pm0.014}$(15)	$0.964_{\pm0.002}$(9)	$0.685_{\pm0.028}$(41)	$0.9_{\pm0.02}$(27)	$0.975_{\pm0.002}$(5)	$0.769_{\pm0.049}$(35)	$0.852_{\pm0.044}$(29)	$0.958_{\pm0.004}$(13)	$0.965_{\pm0.003}$(7)
Stamps	$0.935_{\pm0.016}$(8)	$0.746_{\pm0.021}$(33)	$0.884_{\pm0.011}$(26)	$0.921_{\pm0.03}$(17)	$0.924_{\pm0.022}$(16)	$0.926_{\pm0.014}$(14)	$0.938_{\pm0.016}$(6)	$0.955_{\pm0.011}$(1)	$0.914_{\pm0.04}$(19)	$0.911_{\pm0.032}$(20)	$0.855_{\pm0.029}$(28)	$0.936_{\pm0.02}$(7)
vertebral	$0.487_{\pm0.033}$(16)	$0.455_{\pm0.034}$(24)	$0.416_{\pm0.01}$(33)	$0.411_{\pm0.011}$(35)	$0.489_{\pm0.016}$(14)	$0.362_{\pm0.038}$(40)	$0.426_{\pm0.009}$(30)	$0.412_{\pm0.024}$(34)	$0.387_{\pm0.083}$(38)	$0.386_{\pm0.066}$(39)	$0.469_{\pm0.026}$(20)	$0.502_{\pm0.022}$(13)
WBC	$0.979_{\pm0.008}$(16)	$0.971_{\pm0.006}$(24)	$0.993_{\pm0.002}$(2)	$0.709_{\pm0.248}$(39)	$0.978_{\pm0.011}$(18)	$0.988_{\pm0.005}$(6)	$0.994_{\pm0.003}$(1)	$0.982_{\pm0.008}$(13)	$0.99_{\pm0.006}$(5)	$0.97_{\pm0.012}$(25)	$0.986_{\pm0.006}$(8)	$0.985_{\pm0.009}$(11)
glass	$0.864_{\pm0.016}$(7)	$0.784_{\pm0.043}$(20)	$0.715_{\pm0.025}$(31)	$0.753_{\pm0.067}$(25)	$0.77_{\pm0.027}$(21)	$0.828_{\pm0.027}$(15)	$0.806_{\pm0.014}$(16)	$0.85_{\pm0.021}$(11)	$0.647_{\pm0.052}$(37)	$0.623_{\pm0.101}$(41)	$0.79_{\pm0.008}$(19)	$0.687_{\pm0.052}$(33)
WPBC	$0.527_{\pm0.014}$(18)	$0.465_{\pm0.038}$(44)	$0.5_{\pm0.016}$(31)	$0.543_{\pm0.022}$(12)	$0.508_{\pm0.025}$(26)	$0.578_{\pm0.019}$(1)	$0.549_{\pm0.017}$(9)	$0.558_{\pm0.028}$(6)	$0.495_{\pm0.079}$(34)	$0.543_{\pm0.041}$(12)	$0.515_{\pm0.027}$(21)	$0.528_{\pm0.01}$(17)
Lymphography	$0.991_{\pm0.008}$(8)	$0.957_{\pm0.013}$(36)	$0.994_{\pm0.004}$(3)	$0.989_{\pm0.009}$(17)	$0.984_{\pm0.008}$(23)	$0.993_{\pm0.005}$(4)	$0.995_{\pm0.005}$(2)	$0.989_{\pm0.01}$(17)	$0.957_{\pm0.047}$(36)	$0.642_{\pm0.25}$(39)	$0.983_{\pm0.006}$(24)	$0.99_{\pm0.009}$(14)
wine	$0.965_{\pm0.014}$(8)	$0.325_{\pm0.054}$(42)	$0.74_{\pm0.026}$(29)	$0.961_{\pm0.015}$(11)	$0.982_{\pm0.009}$(2)	$0.933_{\pm0.04}$(19)	$0.934_{\pm0.04}$(18)	$0.973_{\pm0.015}$(6)	$0.875_{\pm0.048}$(25)	$0.913_{\pm0.035}$(21)	$0.983_{\pm0.016}$(1)	$0.962_{\pm0.017}$(10)
Hepatitis	$0.832_{\pm0.033}$(5)	$0.574_{\pm0.057}$(41)	$0.739_{\pm0.032}$(26)	$0.837_{\pm0.024}$(1)	$0.814_{\pm0.048}$(12)	$0.814_{\pm0.034}$(12)	$0.79_{\pm0.049}$(17)	$0.814_{\pm0.031}$(12)	$0.755_{\pm0.047}$(25)	$0.667_{\pm0.076}$(33)	$0.816_{\pm0.063}$(11)	$0.837_{\pm0.024}$(1)
Avg Ranking	11.08	30.42	23.83	18.83	15.17	14.92	11.83	9.75	27.5	27.83	15.92	12.0

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
Pima	$0.741_{\pm0.014}$(3)	$0.721_{\pm0.033}$(9)	$0.638_{\pm0.02}$(29)	$0.568_{\pm0.011}$(38)	$0.665_{\pm0.02}$(24)	$0.681_{\pm0.016}$(20)	$0.536_{\pm0.037}$(40)	$0.596_{\pm0.011}$(34)	$0.607_{\pm0.019}$(31)
breastw	$0.474_{\pm0.121}$(39)	$0.986_{\pm0.005}$(15)	$0.432_{\pm0.015}$(40)	$0.327_{\pm0.023}$(43)	$0.667_{\pm0.034}$(34)	$0.972_{\pm0.005}$(23)	$0.606_{\pm0.081}$(37)	$0.353_{\pm0.021}$(42)	$0.749_{\pm0.004}$(31)
WDBC	$0.453_{\pm0.14}$(42)	$0.988_{\pm0.01}$(16)	$0.881_{\pm0.017}$(37)	$0.942_{\pm0.011}$(34)	$0.952_{\pm0.002}$(33)	$0.974_{\pm0.009}$(28)	$0.645_{\pm0.333}$(40)	$0.975_{\pm0.009}$(26)	$0.981_{\pm0.005}$(22)
Ionosphere	$0.536_{\pm0.02}$(45)	$0.962_{\pm0.018}$(10)	$0.924_{\pm0.004}$(22)	$0.849_{\pm0.008}$(31)	$0.871_{\pm0.011}$(28)	$0.913_{\pm0.005}$(24)	$0.606_{\pm0.048}$(43)	$0.835_{\pm0.011}$(33)	$0.753_{\pm0.009}$(36)
Stamps	$0.887_{\pm0.023}$(23)	$0.931_{\pm0.015}$(12)	$0.653_{\pm0.01}$(37)	$0.434_{\pm0.015}$(44)	$0.742_{\pm0.042}$(34)	$0.706_{\pm0.029}$(36)	$0.516_{\pm0.101}$(42)	$0.447_{\pm0.037}$(43)	$0.885_{\pm0.026}$(24)
vertebral	$0.402_{\pm0.032}$(37)	$0.448_{\pm0.057}$(28)	$0.332_{\pm0.022}$(44)	$0.41_{\pm0.03}$(36)	$0.361_{\pm0.014}$(41)	$0.319_{\pm0.011}$(45)	$0.489_{\pm0.059}$(14)	$0.351_{\pm0.021}$(42)	$0.344_{\pm0.018}$(43)
WBC	$0.354_{\pm0.166}$(43)	$0.978_{\pm0.015}$(18)	$0.942_{\pm0.021}$(28)	$0.901_{\pm0.047}$(34)	$0.876_{\pm0.012}$(36)	$0.978_{\pm0.009}$(18)	$0.529_{\pm0.136}$(41)	$0.921_{\pm0.023}$(31)	$0.993_{\pm0.002}$(2)
glass	$0.857_{\pm0.017}$(8)	$0.855_{\pm0.029}$(9)	$0.796_{\pm0.028}$(18)	$0.752_{\pm0.047}$(26)	$0.736_{\pm0.047}$(28)	$0.851_{\pm0.023}$(10)	$0.412_{\pm0.101}$(45)	$0.725_{\pm0.048}$(29)	$0.638_{\pm0.051}$(38)
WPBC	$0.556_{\pm0.024}$(7)	$0.556_{\pm0.031}$(7)	$0.446_{\pm0.026}$(45)	$0.484_{\pm0.034}$(39)	$0.501_{\pm0.012}$(30)	$0.512_{\pm0.018}$(22)	$0.507_{\pm0.015}$(27)	$0.52_{\pm0.012}$(19)	$0.48_{\pm0.023}$(41)
Lymphography	$0.086_{\pm0.043}$(45)	$0.991_{\pm0.008}$(8)	$0.969_{\pm0.011}$(35)	$0.991_{\pm0.007}$(8)	$0.973_{\pm0.008}$(29)	$0.991_{\pm0.008}$(8)	$0.604_{\pm0.112}$(41)	$0.99_{\pm0.008}$(14)	$0.996_{\pm0.004}$(1)
wine	$0.494_{\pm0.136}$(36)	$0.958_{\pm0.035}$(12)	$0.332_{\pm0.065}$(41)	$0.454_{\pm0.14}$(38)	$0.711_{\pm0.091}$(30)	$0.442_{\pm0.067}$(39)	$0.694_{\pm0.277}$(32)	$0.939_{\pm0.025}$(16)	$0.7_{\pm0.011}$(31)
Hepatitis	$0.611_{\pm0.125}$(38)	$0.817_{\pm0.039}$(10)	$0.624_{\pm0.063}$(36)	$0.715_{\pm0.025}$(29)	$0.637_{\pm0.029}$(34)	$0.783_{\pm0.028}$(19)	$0.6_{\pm0.107}$(39)	$0.805_{\pm0.015}$(16)	$0.622_{\pm0.043}$(37)
Avg Ranking	30.5	12.83	34.33	33.33	31.75	24.33	36.75	28.75	28.08

Table 7: AUPRC results on 12 small datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank).

Dataset	TCMM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
Pima	$0.716_{\pm0.029}$(5)	$0.68_{\pm0.023}$(16)	$0.648_{\pm0.009}$(25)	$0.548_{\pm0.007}$(41)	$0.697_{\pm0.046}$(13)	NAN	$0.665_{\pm0.028}$(21)	$0.578_{\pm0.024}$(38)	$0.502_{\pm0.121}$(43)	$0.636_{\pm0.029}$(28)	$0.594_{\pm0.018}$(35)	$0.657_{\pm0.027}$(24)
breastw	$0.987_{\pm0.007}$(8)	$0.976_{\pm0.011}$(20)	$0.98_{\pm0.007}$(17)	$0.692_{\pm0.026}$(35)	$0.993_{\pm0.001}$(2)	NAN	$0.982_{\pm0.009}$(14)	$0.683_{\pm0.06}$(36)	$0.574_{\pm0.22}$(39)	$0.893_{\pm0.016}$(29)	$0.955_{\pm0.017}$(24)	$0.907_{\pm0.037}$(28)
WDBC	$0.862_{\pm0.074}$(5)	$0.868_{\pm0.091}$(3)	$0.778_{\pm0.217}$(22)	$0.166_{\pm0.105}$(41)	$0.803_{\pm0.122}$(15)	NAN	$0.782_{\pm0.179}$(21)	$0.12_{\pm0.026}$(43)	$0.126_{\pm0.187}$(42)	$0.565_{\pm0.087}$(32)	$0.432_{\pm0.159}$(33)	$0.865_{\pm0.101}$(4)
Ionosphere	$0.983_{\pm0.004}$(2)	$0.959_{\pm0.004}$(15)	$0.946_{\pm0.01}$(18)	$0.595_{\pm0.061}$(44)	$0.806_{\pm0.038}$(34)	NAN	$0.862_{\pm0.044}$(32)	$0.946_{\pm0.005}$(18)	$0.775_{\pm0.126}$(37)	$0.959_{\pm0.007}$(15)	$0.876_{\pm0.022}$(31)	$0.982_{\pm0.004}$(3)
Stamps	$0.651_{\pm0.086}$(7)	$0.601_{\pm0.078}$(15)	$0.654_{\pm0.112}$(5)	$0.274_{\pm0.092}$(39)	$0.486_{\pm0.231}$(28)	NAN	$0.68_{\pm0.109}$(2)	$0.629_{\pm0.081}$(8)	$0.186_{\pm0.094}$(45)	$0.568_{\pm0.094}$(21)	$0.426_{\pm0.09}$(32)	$0.491_{\pm0.128}$(27)
vertebral	$0.311_{\pm0.052}$(1)	$0.2_{\pm0.008}$(27)	$0.233_{\pm0.018}$(13)	$0.238_{\pm0.055}$(9)	$0.248_{\pm0.043}$(7)	NAN	$0.187_{\pm0.017}$(34)	$0.258_{\pm0.015}$(5)	$0.234_{\pm0.117}$(11)	$0.234_{\pm0.027}$(11)	$0.204_{\pm0.023}$(24)	$0.221_{\pm0.007}$(14)
WBC	$0.858_{\pm0.092}$(9)	$0.797_{\pm0.104}$(16)	$0.768_{\pm0.118}$(22)	$0.226_{\pm0.177}$(41)	$0.923_{\pm0.086}$(4)	NAN	$0.832_{\pm0.1}$(14)	$0.173_{\pm0.03}$(42)	$0.234_{\pm0.272}$(40)	$0.462_{\pm0.167}$(33)	$0.615_{\pm0.107}$(29)	$0.66_{\pm0.198}$(27)
glass	$0.355_{\pm0.054}$(2)	$0.217_{\pm0.046}$(27)	$0.242_{\pm0.053}$(20)	$0.096_{\pm0.032}$(45)	$0.2_{\pm0.065}$(32)	NAN	$0.185_{\pm0.069}$(38)	$0.327_{\pm0.049}$(5)	$0.263_{\pm0.122}$(13)	$0.258_{\pm0.092}$(15)	$0.242_{\pm0.079}$(20)	$0.208_{\pm0.056}$(29)
WPBC	$0.418_{\pm0.014}$(4)	$0.385_{\pm0.019}$(30)	$0.365_{\pm0.028}$(41)	$0.379_{\pm0.044}$(34)	$0.41_{\pm0.064}$(7)	NAN	$0.394_{\pm0.053}$(23)	$0.363_{\pm0.014}$(44)	$0.404_{\pm0.073}$(13)	$0.401_{\pm0.019}$(16)	$0.43_{\pm0.051}$(2)	$0.385_{\pm0.02}$(30)
Lymphography	$0.88_{\pm0.11}$(7)	$0.851_{\pm0.161}$(19)	$0.797_{\pm0.243}$(29)	$0.24_{\pm0.13}$(42)	$0.877_{\pm0.112}$(9)	NAN	$0.79_{\pm0.161}$(30)	$0.449_{\pm0.182}$(38)	$0.317_{\pm0.253}$(40)	$0.828_{\pm0.182}$(25)	$0.783_{\pm0.086}$(31)	$0.777_{\pm0.184}$(32)
wine	$0.84_{\pm0.082}$(5)	$0.707_{\pm0.108}$(15)	$0.69_{\pm0.157}$(16)	$0.19_{\pm0.09}$(37)	$0.492_{\pm0.226}$(27)	NAN	$0.504_{\pm0.17}$(26)	$0.224_{\pm0.057}$(36)	$0.172_{\pm0.123}$(38)	$0.752_{\pm0.15}$(12)	$0.246_{\pm0.09}$(34)	$0.573_{\pm0.127}$(22)
Hepatitis	$0.676_{\pm0.065}$(3)	$0.67_{\pm0.058}$(6)	NAN	$0.278_{\pm0.038}$(44)	$0.518_{\pm0.116}$(27)	NAN	$0.568_{\pm0.043}$(21)	$0.446_{\pm0.065}$(33)	$0.331_{\pm0.133}$(41)	$0.653_{\pm0.05}$(10)	$0.532_{\pm0.107}$(26)	$0.652_{\pm0.026}$(11)
Avg Ranking	4.83	17.42	20.73	37.67	17.08	NAN	23.0	28.83	33.5	20.58	26.75	20.92

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
Pima	$0.624_{\pm0.009}$(31)	$0.723_{\pm0.027}$(4)	$0.602_{\pm0.039}$(34)	$0.464_{\pm0.044}$(44)	$0.666_{\pm0.019}$(20)	$0.713_{\pm0.028}$(7)	$0.704_{\pm0.029}$(9)	$0.46_{\pm0.053}$(45)	$0.698_{\pm0.024}$(12)	$0.568_{\pm0.018}$(39)	$0.504_{\pm0.08}$(42)	$0.659_{\pm0.012}$(23)
breastw	$0.722_{\pm0.137}$(34)	$0.98_{\pm0.01}$(17)	$0.942_{\pm0.017}$(26)	$0.833_{\pm0.087}$(31)	$0.942_{\pm0.025}$(26)	$0.986_{\pm0.009}$(11)	$0.982_{\pm0.011}$(14)	$0.337_{\pm0.028}$(45)	$0.96_{\pm0.024}$(23)	$0.989_{\pm0.004}$(4)	$0.465_{\pm0.139}$(41)	$0.988_{\pm0.007}$(6)
WDBC	$0.82_{\pm0.111}$(11)	$0.839_{\pm0.089}$(9)	$0.42_{\pm0.175}$(34)	$0.808_{\pm0.127}$(12)	$0.789_{\pm0.143}$(19)	$0.892_{\pm0.08}$(1)	$0.653_{\pm0.358}$(26)	$0.03_{\pm0.0}$(45)	$0.7_{\pm0.065}$(23)	$0.698_{\pm0.12}$(24)	$0.036_{\pm0.006}$(44)	$0.787_{\pm0.142}$(20)
Ionosphere	$0.974_{\pm0.009}$(8)	$0.982_{\pm0.004}$(3)	$0.918_{\pm0.058}$(26)	$0.943_{\pm0.013}$(23)	$0.968_{\pm0.007}$(12)	$0.981_{\pm0.005}$(5)	$0.638_{\pm0.101}$(42)	$0.728_{\pm0.095}$(41)	$0.943_{\pm0.021}$(23)	$0.967_{\pm0.004}$(13)	$0.755_{\pm0.081}$(39)	$0.924_{\pm0.019}$(25)
Stamps	$0.461_{\pm0.12}$(31)	$0.665_{\pm0.083}$(3)	$0.585_{\pm0.184}$(16)	$0.389_{\pm0.235}$(35)	$0.51_{\pm0.062}$(26)	$0.652_{\pm0.1}$(6)	$0.584_{\pm0.082}$(17)	$0.395_{\pm0.262}$(34)	$0.542_{\pm0.047}$(23)	$0.272_{\pm0.022}$(40)	$0.475_{\pm0.207}$(30)	$0.608_{\pm0.097}$(11)
vertebral	$0.285_{\pm0.042}$(2)	$0.2_{\pm0.005}$(27)	$0.204_{\pm0.035}$(24)	$0.211_{\pm0.027}$(19)	$0.248_{\pm0.04}$(7)	$0.199_{\pm0.013}$(29)	$0.215_{\pm0.057}$(15)	$0.277_{\pm0.065}$(3)	$0.257_{\pm0.055}$(6)	$0.205_{\pm0.012}$(23)	$0.276_{\pm0.07}$(4)	$0.212_{\pm0.01}$(17)
WBC	$0.35_{\pm0.21}$(36)	$0.771_{\pm0.105}$(21)	$0.612_{\pm0.192}$(30)	$0.686_{\pm0.095}$(26)	$0.537_{\pm0.153}$(31)	$0.779_{\pm0.101}$(20)	$0.767_{\pm0.148}$(23)	$0.049_{\pm0.001}$(45)	$0.711_{\pm0.151}$(25)	$0.866_{\pm0.099}$(6)	$0.05_{\pm0.003}$(44)	$0.866_{\pm0.097}$(6)
glass	$0.243_{\pm0.039}$(19)	$0.302_{\pm0.05}$(7)	$0.223_{\pm0.07}$(26)	$0.2_{\pm0.1}$(32)	$0.447_{\pm0.072}$(1)	$0.345_{\pm0.051}$(3)	$0.266_{\pm0.045}$(12)	$0.198_{\pm0.079}$(34)	$0.241_{\pm0.065}$(22)	$0.267_{\pm0.061}$(11)	$0.212_{\pm0.075}$(28)	$0.189_{\pm0.015}$(37)
WPBC	$0.395_{\pm0.029}$(22)	$0.409_{\pm0.021}$(10)	$0.43_{\pm0.041}$(2)	$0.434_{\pm0.044}$(1)	$0.406_{\pm0.033}$(12)	$0.412_{\pm0.019}$(5)	$0.387_{\pm0.052}$(28)	$0.377_{\pm0.034}$(36)	$0.403_{\pm0.042}$(14)	$0.401_{\pm0.007}$(16)	$0.396_{\pm0.048}$(20)	$0.383_{\pm0.018}$(32)
Lymphography	$0.754_{\pm0.166}$(33)	$0.832_{\pm0.153}$(21)	$0.748_{\pm0.11}$(34)	$0.823_{\pm0.149}$(26)	$0.873_{\pm0.062}$(13)	$0.852_{\pm0.079}$(18)	$0.9_{\pm0.071}$(6)	$0.1_{\pm0.115}$(44)	$0.798_{\pm0.266}$(28)	$0.853_{\pm0.14}$(17)	$0.126_{\pm0.144}$(43)	$0.91_{\pm0.105}$(5)
wine	$0.648_{\pm0.128}$(19)	$0.809_{\pm0.084}$(7)	$0.31_{\pm0.192}$(31)	$0.768_{\pm0.082}$(10)	$0.506_{\pm0.073}$(25)	$0.843_{\pm0.091}$(4)	$0.459_{\pm0.295}$(28)	$0.094_{\pm0.015}$(43)	$0.572_{\pm0.197}$(23)	$0.88_{\pm0.064}$(1)	$0.089_{\pm0.009}$(44)	$0.651_{\pm0.095}$(18)
Hepatitis	$0.591_{\pm0.079}$(17)	$0.66_{\pm0.057}$(9)	$0.549_{\pm0.083}$(23)	$0.666_{\pm0.052}$(8)	$0.458_{\pm0.089}$(31)	$0.547_{\pm0.122}$(24)	$0.576_{\pm0.051}$(20)	$0.388_{\pm0.076}$(37)	$0.385_{\pm0.101}$(38)	$0.594_{\pm0.083}$(16)	$0.321_{\pm0.057}$(43)	$0.675_{\pm0.058}$(5)
Avg Ranking	21.92	11.5	25.5	22.25	18.58	11.08	20.0	37.67	21.67	17.5	35.17	17.08

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
Pima	$0.691_{\pm0.024}$(14)	$0.662_{\pm0.02}$(22)	$0.627_{\pm0.022}$(30)	$0.668_{\pm0.019}$(18)	$0.7_{\pm0.025}$(10)	$0.736_{\pm0.024}$(2)	$0.715_{\pm0.018}$(6)	$0.732_{\pm0.027}$(3)	$0.633_{\pm0.054}$(29)	$0.619_{\pm0.068}$(32)	$0.677_{\pm0.025}$(17)	$0.691_{\pm0.025}$(14)
breastw	$0.987_{\pm0.007}$(8)	$0.976_{\pm0.004}$(20)	$0.992_{\pm0.001}$(3)	$0.558_{\pm0.216}$(40)	$0.979_{\pm0.008}$(19)	$0.989_{\pm0.003}$(4)	$0.995_{\pm0.001}$(1)	$0.987_{\pm0.008}$(8)	$0.798_{\pm0.044}$(32)	$0.97_{\pm0.019}$(22)	$0.988_{\pm0.003}$(6)	$0.986_{\pm0.01}$(11)
WDBC	$0.807_{\pm0.095}$(13)	$0.414_{\pm0.038}$(35)	$0.648_{\pm0.07}$(27)	$0.883_{\pm0.053}$(2)	$0.797_{\pm0.11}$(17)	$0.801_{\pm0.075}$(16)	$0.792_{\pm0.094}$(18)	$0.861_{\pm0.074}$(7)	$0.822_{\pm0.144}$(10)	$0.58_{\pm0.191}$(31)	$0.598_{\pm0.085}$(29)	$0.843_{\pm0.095}$(8)
Ionosphere	$0.966_{\pm0.01}$(14)	$0.944_{\pm0.003}$(22)	$0.766_{\pm0.001}$(38)	$0.958_{\pm0.011}$(17)	$0.974_{\pm0.002}$(8)	$0.61_{\pm0.023}$(43)	$0.907_{\pm0.018}$(27)	$0.979_{\pm0.002}$(6)	$0.801_{\pm0.046}$(36)	$0.837_{\pm0.07}$(33)	$0.969_{\pm0.003}$(10)	$0.975_{\pm0.003}$(7)
Stamps	$0.664_{\pm0.083}$(4)	$0.325_{\pm0.015}$(38)	$0.516_{\pm0.043}$(25)	$0.579_{\pm0.108}$(18)	$0.603_{\pm0.091}$(14)	$0.56_{\pm0.056}$(22)	$0.611_{\pm0.1}$(10)	$0.688_{\pm0.069}$(1)	$0.577_{\pm0.075}$(19)	$0.54_{\pm0.075}$(24)	$0.424_{\pm0.072}$(33)	$0.619_{\pm0.091}$(9)
vertebral	$0.211_{\pm0.012}$(19)	$0.208_{\pm0.019}$(22)	$0.191_{\pm0.004}$(31)	$0.19_{\pm0.009}$(32)	$0.21_{\pm0.005}$(21)	$0.174_{\pm0.011}$(40)	$0.189_{\pm0.002}$(33)	$0.184_{\pm0.007}$(38)	$0.186_{\pm0.027}$(35)	$0.18_{\pm0.018}$(39)	$0.204_{\pm0.01}$(24)	$0.213_{\pm0.007}$(16)
WBC	$0.797_{\pm0.087}$(16)	$0.805_{\pm0.024}$(15)	$0.925_{\pm0.024}$(3)	$0.319_{\pm0.312}$(37)	$0.78_{\pm0.118}$(19)	$0.861_{\pm0.064}$(8)	$0.956_{\pm0.022}$(1)	$0.849_{\pm0.083}$(12)	$0.912_{\pm0.058}$(5)	$0.619_{\pm0.107}$(28)	$0.856_{\pm0.07}$(10)	$0.85_{\pm0.115}$(11)
glass	$0.268_{\pm0.015}$(10)	$0.241_{\pm0.062}$(22)	$0.255_{\pm0.013}$(16)	$0.204_{\pm0.081}$(31)	$0.236_{\pm0.061}$(24)	$0.27_{\pm0.055}$(9)	$0.208_{\pm0.024}$(29)	$0.299_{\pm0.041}$(8)	$0.169_{\pm0.05}$(40)	$0.118_{\pm0.029}$(43)	$0.197_{\pm0.018}$(35)	$0.226_{\pm0.051}$(25)
WPBC	$0.391_{\pm0.01}$(24)	$0.364_{\pm0.018}$(43)	$0.368_{\pm0.012}$(39)	$0.403_{\pm0.02}$(14)	$0.389_{\pm0.024}$(26)	$0.412_{\pm0.017}$(5)	$0.401_{\pm0.017}$(16)	$0.41_{\pm0.024}$(7)	$0.386_{\pm0.042}$(29)	$0.41_{\pm0.022}$(7)	$0.396_{\pm0.021}$(20)	$0.39_{\pm0.013}$(25)
Lymphography	$0.877_{\pm0.112}$(9)	$0.673_{\pm0.145}$(36)	$0.937_{\pm0.048}$(3)	$0.849_{\pm0.13}$(20)	$0.832_{\pm0.019}$(21)	$0.929_{\pm0.046}$(4)	$0.948_{\pm0.045}$(2)	$0.867_{\pm0.112}$(16)	$0.811_{\pm0.145}$(27)	$0.264_{\pm0.236}$(41)	$0.829_{\pm0.041}$(24)	$0.832_{\pm0.153}$(21)
wine	$0.778_{\pm0.076}$(8)	$0.111_{\pm0.008}$(42)	$0.327_{\pm0.048}$(30)	$0.748_{\pm0.083}$(13)	$0.853_{\pm0.081}$(3)	$0.674_{\pm0.119}$(17)	$0.715_{\pm0.181}$(14)	$0.826_{\pm0.079}$(6)	$0.557_{\pm0.115}$(24)	$0.631_{\pm0.116}$(20)	$0.875_{\pm0.125}$(2)	$0.773_{\pm0.087}$(9)
Hepatitis	$0.68_{\pm0.064}$(1)	$0.323_{\pm0.032}$(42)	$0.461_{\pm0.05}$(30)	$0.679_{\pm0.061}$(2)	$0.607_{\pm0.047}$(15)	$0.59_{\pm0.066}$(19)	$0.56_{\pm0.078}$(22)	$0.614_{\pm0.043}$(14)	$0.591_{\pm0.076}$(17)	$0.469_{\pm0.124}$(29)	$0.619_{\pm0.114}$(13)	$0.676_{\pm0.053}$(3)
Avg Ranking	11.67	29.92	22.92	20.33	16.42	15.75	14.92	10.5	25.25	29.08	18.58	13.25

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
Pima	$0.749_{\pm0.019}$(1)	$0.707_{\pm0.048}$(8)	$0.639_{\pm0.026}$(26)	$0.588_{\pm0.019}$(36)	$0.638_{\pm0.025}$(27)	$0.668_{\pm0.021}$(18)	$0.556_{\pm0.02}$(40)	$0.58_{\pm0.018}$(37)	$0.612_{\pm0.023}$(33)
breastw	$0.639_{\pm0.09}$(37)	$0.981_{\pm0.009}$(16)	$0.44_{\pm0.006}$(42)	$0.399_{\pm0.007}$(44)	$0.596_{\pm0.036}$(38)	$0.945_{\pm0.01}$(25)	$0.744_{\pm0.06}$(33)	$0.41_{\pm0.007}$(43)	$0.845_{\pm0.003}$(30)
WDBC	$0.171_{\pm0.056}$(40)	$0.807_{\pm0.151}$(13)	$0.281_{\pm0.032}$(39)	$0.374_{\pm0.07}$(37)	$0.392_{\pm0.038}$(36)	$0.596_{\pm0.099}$(30)	$0.351_{\pm0.315}$(38)	$0.599_{\pm0.106}$(28)	$0.669_{\pm0.058}$(25)
Ionosphere	$0.552_{\pm0.025}$(45)	$0.969_{\pm0.014}$(10)	$0.946_{\pm0.005}$(18)	$0.901_{\pm0.008}$(29)	$0.906_{\pm0.008}$(28)	$0.945_{\pm0.004}$(21)	$0.744_{\pm0.032}$(40)	$0.88_{\pm0.013}$(30)	$0.805_{\pm0.004}$(35)
Stamps	$0.57_{\pm0.037}$(20)	$0.607_{\pm0.043}$(12)	$0.27_{\pm0.018}$(41)	$0.234_{\pm0.017}$(42)	$0.34_{\pm0.029}$(36)	$0.332_{\pm0.035}$(37)	$0.234_{\pm0.039}$(42)	$0.232_{\pm0.018}$(44)	$0.484_{\pm0.066}$(29)
vertebral	$0.185_{\pm0.013}$(37)	$0.198_{\pm0.022}$(30)	$0.166_{\pm0.005}$(44)	$0.186_{\pm0.007}$(35)	$0.172_{\pm0.004}$(41)	$0.161_{\pm0.002}$(45)	$0.212_{\pm0.021}$(17)	$0.169_{\pm0.005}$(42)	$0.167_{\pm0.005}$(43)
WBC	$0.153_{\pm0.088}$(43)	$0.787_{\pm0.128}$(18)	$0.524_{\pm0.137}$(32)	$0.382_{\pm0.102}$(35)	$0.309_{\pm0.012}$(38)	$0.751_{\pm0.119}$(24)	$0.261_{\pm0.175}$(39)	$0.391_{\pm0.062}$(34)	$0.95_{\pm0.016}$(2)
glass	$0.327_{\pm0.084}$(5)	$0.26_{\pm0.044}$(14)	$0.244_{\pm0.052}$(18)	$0.183_{\pm0.041}$(39)	$0.196_{\pm0.056}$(36)	$0.254_{\pm0.019}$(17)	$0.107_{\pm0.044}$(44)	$0.167_{\pm0.037}$(42)	$0.169_{\pm0.018}$(40)
WPBC	$0.401_{\pm0.019}$(16)	$0.408_{\pm0.039}$(11)	$0.351_{\pm0.024}$(45)	$0.365_{\pm0.018}$(41)	$0.368_{\pm0.01}$(39)	$0.375_{\pm0.013}$(37)	$0.381_{\pm0.006}$(33)	$0.378_{\pm0.011}$(35)	$0.369_{\pm0.018}$(38)
Lymphography	$0.05_{\pm0.004}$(45)	$0.872_{\pm0.106}$(14)	$0.67_{\pm0.05}$(37)	$0.875_{\pm0.103}$(12)	$0.697_{\pm0.103}$(35)	$0.872_{\pm0.106}$(14)	$0.351_{\pm0.184}$(39)	$0.878_{\pm0.094}$(8)	$0.963_{\pm0.041}$(1)
wine	$0.294_{\pm0.117}$(32)	$0.76_{\pm0.135}$(11)	$0.113_{\pm0.01}$(41)	$0.138_{\pm0.032}$(40)	$0.234_{\pm0.057}$(35)	$0.142_{\pm0.018}$(39)	$0.437_{\pm0.305}$(29)	$0.596_{\pm0.096}$(21)	$0.271_{\pm0.013}$(33)
Hepatitis	$0.405_{\pm0.101}$(35)	$0.624_{\pm0.07}$(12)	$0.357_{\pm0.054}$(39)	$0.436_{\pm0.052}$(34)	$0.349_{\pm0.025}$(40)	$0.505_{\pm0.054}$(28)	$0.449_{\pm0.12}$(32)	$0.546_{\pm0.052}$(25)	$0.392_{\pm0.051}$(36)
Avg Ranking	29.67	14.08	35.17	35.33	35.75	27.92	35.5	32.42	28.75

Table 8: AUROC results on 15 medium datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank in terms of mean among all anomaly detectors).

Dataset	TCMM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
annthyroid	$0.918_{\pm 0.032}$(8)	$0.856_{\pm 0.042}$(23)	$0.86_{\pm 0.014}$(21)	$0.575_{\pm 0.019}$(43)	$0.806_{\pm 0.055}$(27)	$0.602_{\pm 0.167}$(42)	$0.857_{\pm 0.042}$(22)	$0.778_{\pm 0.045}$(30)	$0.876_{\pm 0.035}$(16)	$0.982_{\pm 0.001}$(1)	$0.771_{\pm 0.012}$(31)	$0.956_{\pm 0.016}$(2)
pendigits	$0.983_{\pm 0.002}$(7)	$0.977_{\pm 0.016}$(10)	$0.955_{\pm 0.012}$(18)	$0.552_{\pm 0.051}$(41)	$0.704_{\pm 0.289}$(36)	$0.688_{\pm 0.149}$(37)	$0.871_{\pm 0.092}$(27)	$0.982_{\pm 0.006}$(8)	$0.774_{\pm 0.142}$(34)	$0.972_{\pm 0.011}$(12)	$0.83_{\pm 0.018}$(31)	$0.993_{\pm 0.001}$(5)
satellite	$0.825_{\pm 0.008}$(10)	$0.796_{\pm 0.009}$(18)	$0.769_{\pm 0.006}$(22)	$0.504_{\pm 0.025}$(45)	$0.641_{\pm 0.137}$(36)	$0.689_{\pm 0.071}$(32)	$0.675_{\pm 0.043}$(33)	$0.778_{\pm 0.006}$(20)	$0.728_{\pm 0.095}$(27)	$0.809_{\pm 0.007}$(14)	$0.772_{\pm 0.006}$(21)	$0.795_{\pm 0.008}$(19)
landsat	$0.619_{\pm 0.011}$(11)	$0.57_{\pm 0.007}$(21)	$0.586_{\pm 0.008}$(16)	$0.485_{\pm 0.019}$(32)	$0.437_{\pm 0.168}$(41)	$0.508_{\pm 0.078}$(29)	$0.433_{\pm 0.066}$(42)	$0.577_{\pm 0.006}$(19)	$0.588_{\pm 0.023}$(15)	$0.575_{\pm 0.027}$(20)	$0.521_{\pm 0.01}$(26)	$0.46_{\pm 0.021}$(37)
satimage-2	$0.998_{\pm 0.001}$(6)	$0.999_{\pm 0.0}$(1)	$0.987_{\pm 0.0}$(23)	$0.532_{\pm 0.044}$(46)	$0.927_{\pm 0.028}$(35)	$0.787_{\pm 0.145}$(40)	$0.974_{\pm 0.023}$(30)	$0.997_{\pm 0.001}$(10)	$0.818_{\pm 0.126}$(39)	$0.988_{\pm 0.001}$(21)	$0.975_{\pm 0.009}$(29)	$0.995_{\pm 0.001}$(16)
PageBlocks	$0.96_{\pm 0.005}$(6)	$0.95_{\pm 0.015}$(12)	$0.964_{\pm 0.008}$(2)	$0.581_{\pm 0.048}$(44)	$0.859_{\pm 0.029}$(33)	$0.765_{\pm 0.09}$(39)	$0.926_{\pm 0.021}$(22)	$0.935_{\pm 0.001}$(19)	$0.947_{\pm 0.024}$(15)	$0.962_{\pm 0.006}$(3)	$0.875_{\pm 0.016}$(30)	$0.959_{\pm 0.006}$(8)
Wilt	$0.939_{\pm 0.012}$(3)	$0.55_{\pm 0.033}$(19)	$0.459_{\pm 0.019}$(29)	$0.468_{\pm 0.017}$(27)	$0.377_{\pm 0.079}$(38)	$0.574_{\pm 0.091}$(17)	$0.353_{\pm 0.048}$(40)	$0.361_{\pm 0.014}$(39)	$0.486_{\pm 0.107}$(25)	$0.863_{\pm 0.008}$(4)	$0.506_{\pm 0.014}$(23)	$0.642_{\pm 0.022}$(12)
thyroid	$0.982_{\pm 0.009}$(13)	$0.98_{\pm 0.005}$(16)	$0.99_{\pm 0.002}$(2)	$0.594_{\pm 0.047}$(42)	$0.928_{\pm 0.126}$(33)	$0.737_{\pm 0.194}$(41)	$0.985_{\pm 0.003}$(8)	$0.984_{\pm 0.003}$(10)	$0.921_{\pm 0.024}$(34)	$0.993_{\pm 0.001}$(1)	$0.881_{\pm 0.038}$(38)	$0.945_{\pm 0.043}$(28)
Waveform	$0.738_{\pm 0.064}$(8)	$0.687_{\pm 0.032}$(19)	$0.686_{\pm 0.01}$(20)	$0.51_{\pm 0.036}$(41)	$0.684_{\pm 0.046}$(21)	$0.492_{\pm 0.015}$(42)	$0.59_{\pm 0.133}$(34)	$0.722_{\pm 0.009}$(11)	$0.68_{\pm 0.052}$(22)	$0.62_{\pm 0.026}$(29)	$0.523_{\pm 0.026}$(39)	$0.588_{\pm 0.01}$(35)
Cardiotocography	$0.829_{\pm 0.012}$(2)	$0.74_{\pm 0.016}$(18)	$0.756_{\pm 0.052}$(15)	$0.557_{\pm 0.039}$(38)	$0.742_{\pm 0.125}$(17)	*NAN*	$0.799_{\pm 0.101}$(7)	$0.635_{\pm 0.019}$(31)	$0.44_{\pm 0.172}$(43)	$0.738_{\pm 0.027}$(19)	$0.574_{\pm 0.016}$(36)	$0.756_{\pm 0.013}$(15)
fault	$0.777_{\pm 0.032}$(6)	$0.737_{\pm 0.011}$(8)	$0.633_{\pm 0.012}$(22)	$0.501_{\pm 0.019}$(41)	$0.528_{\pm 0.067}$(36)	*NAN*	$0.542_{\pm 0.048}$(35)	$0.718_{\pm 0.019}$(10)	$0.621_{\pm 0.043}$(25)	$0.695_{\pm 0.017}$(13)	$0.618_{\pm 0.018}$(27)	$0.701_{\pm 0.005}$(11)
cardio	$0.956_{\pm 0.008}$(5)	$0.93_{\pm 0.028}$(17)	$0.958_{\pm 0.007}$(3)	$0.56_{\pm 0.025}$(40)	$0.907_{\pm 0.058}$(22)	*NAN*	$0.938_{\pm 0.029}$(14)	$0.951_{\pm 0.008}$(9)	$0.516_{\pm 0.236}$(41)	$0.908_{\pm 0.019}$(21)	$0.73_{\pm 0.006}$(36)	$0.934_{\pm 0.008}$(16)
letter	$0.891_{\pm 0.012}$(6)	$0.802_{\pm 0.008}$(19)	$0.615_{\pm 0.017}$(31)	$0.507_{\pm 0.041}$(41)	$0.501_{\pm 0.035}$(43)	*NAN*	$0.507_{\pm 0.06}$(41)	$0.667_{\pm 0.02}$(27)	$0.649_{\pm 0.051}$(28)	$0.88_{\pm 0.006}$(7)	$0.642_{\pm 0.045}$(29)	$0.868_{\pm 0.005}$(10)
yeast	$0.503_{\pm 0.026}$(4)	$0.461_{\pm 0.021}$(22)	$0.447_{\pm 0.024}$(28)	$0.481_{\pm 0.014}$(14)	$0.422_{\pm 0.061}$(36)	*NAN*	$0.43_{\pm 0.032}$(33)	$0.402_{\pm 0.021}$(41)	$0.503_{\pm 0.041}$(4)	$0.468_{\pm 0.025}$(19)	$0.493_{\pm 0.02}$(11)	$0.474_{\pm 0.027}$(17)
vowels	$0.97_{\pm 0.008}$(7)	$0.934_{\pm 0.009}$(15)	$0.709_{\pm 0.028}$(31)	$0.502_{\pm 0.078}$(42)	$0.522_{\pm 0.031}$(40)	*NAN*	$0.576_{\pm 0.085}$(38)	$0.814_{\pm 0.018}$(26)	$0.519_{\pm 0.152}$(41)	$0.978_{\pm 0.006}$(5)	$0.734_{\pm 0.043}$(29)	$0.964_{\pm 0.006}$(9)
Avg Ranking	6.8	15.87	18.87	38.47	32.93	35.44	28.4	20.67	27.27	12.6	29.07	16.0

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
annthyroid	$0.909_{\pm 0.057}$(10)	$0.939_{\pm 0.005}$(4)	$0.67_{\pm 0.088}$(38)	$0.654_{\pm 0.042}$(39)	$0.794_{\pm 0.023}$(28)	$0.886_{\pm 0.027}$(13)	$0.89_{\pm 0.044}$(11)	$0.681_{\pm 0.032}$(37)	$0.946_{\pm 0.012}$(3)	$0.933_{\pm 0.004}$(5)	$0.723_{\pm 0.027}$(35)	$0.875_{\pm 0.022}$(18)
pendigits	$0.979_{\pm 0.018}$(9)	$0.999_{\pm 0.0}$(1)	$0.631_{\pm 0.247}$(39)	$0.262_{\pm 0.224}$(46)	$0.966_{\pm 0.022}$(14)	$0.999_{\pm 0.0}$(1)	$0.976_{\pm 0.013}$(11)	$0.779_{\pm 0.068}$(33)	$0.821_{\pm 0.068}$(32)	$0.923_{\pm 0.018}$(24)	$0.748_{\pm 0.104}$(35)	$0.947_{\pm 0.005}$(19)
satellite	$0.755_{\pm 0.051}$(26)	$0.878_{\pm 0.002}$(3)	$0.813_{\pm 0.01}$(12)	$0.725_{\pm 0.023}$(28)	$0.886_{\pm 0.008}$(1)	$0.878_{\pm 0.003}$(3)	$0.767_{\pm 0.062}$(23)	$0.69_{\pm 0.011}$(30)	$0.69_{\pm 0.036}$(30)	$0.881_{\pm 0.006}$(2)	$0.65_{\pm 0.016}$(35)	$0.762_{\pm 0.002}$(24)
landsat	$0.492_{\pm 0.013}$(31)	$0.774_{\pm 0.005}$(2)	$0.618_{\pm 0.06}$(12)	$0.539_{\pm 0.01}$(23)	$0.741_{\pm 0.006}$(4)	$0.783_{\pm 0.006}$(1)	$0.554_{\pm 0.066}$(22)	$0.523_{\pm 0.025}$(25)	$0.474_{\pm 0.012}$(34)	$0.71_{\pm 0.004}$(6)	$0.454_{\pm 0.023}$(40)	$0.581_{\pm 0.003}$(17)
satimage-2	$0.974_{\pm 0.017}$(30)	$0.999_{\pm 0.0}$(1)	$0.982_{\pm 0.009}$(26)	$0.992_{\pm 0.001}$(20)	$0.997_{\pm 0.002}$(10)	$0.998_{\pm 0.001}$(6)	$0.988_{\pm 0.017}$(21)	$0.935_{\pm 0.007}$(34)	$0.959_{\pm 0.014}$(33)	$0.998_{\pm 0.001}$(6)	$0.892_{\pm 0.018}$(38)	$0.985_{\pm 0.001}$(25)
PageBlocks	$0.924_{\pm 0.031}$(23)	$0.962_{\pm 0.001}$(3)	$0.751_{\pm 0.097}$(40)	$0.784_{\pm 0.034}$(36)	$0.931_{\pm 0.01}$(20)	$0.938_{\pm 0.003}$(18)	$0.96_{\pm 0.002}$(6)	$0.642_{\pm 0.056}$(43)	$0.901_{\pm 0.025}$(26)	$0.873_{\pm 0.014}$(31)	$0.811_{\pm 0.06}$(35)	$0.951_{\pm 0.003}$(11)
Wilt	$0.951_{\pm 0.012}$(2)	$0.661_{\pm 0.016}$(10)	$0.446_{\pm 0.064}$(30)	$0.597_{\pm 0.038}$(15)	$0.75_{\pm 0.048}$(7)	$0.512_{\pm 0.044}$(22)	$0.562_{\pm 0.216}$(18)	$0.484_{\pm 0.046}$(26)	$0.76_{\pm 0.07}$(6)	$0.653_{\pm 0.014}$(11)	$0.426_{\pm 0.071}$(34)	$0.446_{\pm 0.006}$(30)
thyroid	$0.887_{\pm 0.118}$(37)	$0.987_{\pm 0.002}$(5)	$0.944_{\pm 0.044}$(29)	$0.563_{\pm 0.088}$(43)	$0.94_{\pm 0.019}$(32)	$0.98_{\pm 0.001}$(16)	$0.978_{\pm 0.011}$(19)	$0.816_{\pm 0.038}$(40)	$0.987_{\pm 0.007}$(5)	$0.944_{\pm 0.018}$(29)	$0.95_{\pm 0.02}$(26)	$0.989_{\pm 0.004}$(4)
Waveform	$0.676_{\pm 0.024}$(23)	$0.755_{\pm 0.085}$(5)	$0.752_{\pm 0.079}$(6)	$0.456_{\pm 0.097}$(45)	$0.692_{\pm 0.036}$(18)	$0.76_{\pm 0.012}$(4)	$0.793_{\pm 0.121}$(1)	$0.458_{\pm 0.049}$(44)	$0.63_{\pm 0.034}$(27)	$0.477_{\pm 0.011}$(43)	$0.443_{\pm 0.051}$(46)	$0.699_{\pm 0.015}$(17)
Cardiotocography	$0.733_{\pm 0.005}$(20)	$0.765_{\pm 0.001}$(12)	$0.679_{\pm 0.06}$(28)	$0.273_{\pm 0.056}$(45)	$0.69_{\pm 0.013}$(27)	$0.808_{\pm 0.015}$(4)	$0.773_{\pm 0.035}$(10)	$0.599_{\pm 0.045}$(33)	$0.73_{\pm 0.038}$(21)	$0.582_{\pm 0.026}$(34)	$0.568_{\pm 0.088}$(37)	$0.832_{\pm 0.005}$(1)
fault	$0.671_{\pm 0.031}$(17)	$0.811_{\pm 0.008}$(2)	$0.613_{\pm 0.07}$(29)	$0.665_{\pm 0.039}$(18)	$0.781_{\pm 0.007}$(5)	$0.807_{\pm 0.012}$(3)	$0.7_{\pm 0.035}$(12)	$0.469_{\pm 0.06}$(43)	$0.511_{\pm 0.031}$(39)	$0.799_{\pm 0.007}$(4)	$0.467_{\pm 0.06}$(44)	$0.618_{\pm 0.022}$(27)
cardio	$0.848_{\pm 0.046}$(30)	$0.945_{\pm 0.005}$(12)	$0.883_{\pm 0.039}$(24)	$0.166_{\pm 0.048}$(45)	$0.879_{\pm 0.026}$(26)	$0.958_{\pm 0.006}$(3)	$0.947_{\pm 0.013}$(10)	$0.796_{\pm 0.078}$(34)	$0.909_{\pm 0.025}$(20)	$0.839_{\pm 0.036}$(33)	$0.794_{\pm 0.053}$(35)	$0.971_{\pm 0.002}$(1)
letter	$0.846_{\pm 0.005}$(15)	$0.903_{\pm 0.009}$(5)	$0.751_{\pm 0.057}$(23)	$0.764_{\pm 0.018}$(21)	$0.925_{\pm 0.009}$(2)	$0.927_{\pm 0.008}$(1)	$0.86_{\pm 0.028}$(12)	$0.385_{\pm 0.011}$(45)	$0.73_{\pm 0.044}$(24)	$0.909_{\pm 0.006}$(4)	$0.389_{\pm 0.023}$(44)	$0.615_{\pm 0.025}$(31)
yeast	$0.516_{\pm 0.038}$(2)	$0.46_{\pm 0.021}$(24)	$0.489_{\pm 0.044}$(12)	$0.592_{\pm 0.016}$(1)	$0.501_{\pm 0.027}$(7)	$0.457_{\pm 0.019}$(26)	$0.424_{\pm 0.028}$(34)	$0.495_{\pm 0.039}$(9)	$0.471_{\pm 0.02}$(18)	$0.513_{\pm 0.009}$(3)	$0.497_{\pm 0.041}$(8)	$0.459_{\pm 0.016}$(25)
vowels	$0.984_{\pm 0.007}$(2)	$0.981_{\pm 0.005}$(4)	$0.7_{\pm 0.072}$(33)	$0.831_{\pm 0.038}$(24)	$0.985_{\pm 0.005}$(1)	$0.984_{\pm 0.005}$(2)	$0.964_{\pm 0.007}$(9)	$0.125_{\pm 0.082}$(45)	$0.86_{\pm 0.052}$(23)	$0.966_{\pm 0.003}$(8)	$0.151_{\pm 0.1}$(44)	$0.636_{\pm 0.037}$(35)
Avg Ranking	18.47	6.2	25.4	29.93	13.47	8.2	14.6	34.73	22.73	16.2	35.73	19.0

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
annthyroid	$0.888_{\pm 0.033}$(12)	$0.624_{\pm 0.003}$(41)	$0.788_{\pm 0.001}$(29)	$0.923_{\pm 0.024}$(6)	$0.834_{\pm 0.029}$(25)	$0.71_{\pm 0.058}$(36)	$0.912_{\pm 0.011}$(9)	$0.88_{\pm 0.026}$(15)	$0.748_{\pm 0.042}$(33)	$0.736_{\pm 0.06}$(34)	$0.921_{\pm 0.005}$(7)	$0.875_{\pm 0.025}$(18)
pendigits	$0.959_{\pm 0.018}$(16)	$0.552_{\pm 0.007}$(41)	$0.929_{\pm 0.001}$(22)	$0.997_{\pm 0.002}$(4)	$0.847_{\pm 0.005}$(29)	$0.937_{\pm 0.001}$(20)	$0.971_{\pm 0.006}$(13)	$0.999_{\pm 0.0}$(1)	$0.896_{\pm 0.034}$(26)	$0.93_{\pm 0.021}$(21)	$0.833_{\pm 0.008}$(30)	$0.966_{\pm 0.004}$(14)
satellite	$0.852_{\pm 0.025}$(8)	$0.577_{\pm 0.002}$(39)	$0.584_{\pm 0.004}$(38)	$0.846_{\pm 0.005}$(9)	$0.802_{\pm 0.002}$(16)	$0.867_{\pm 0.004}$(6)	$0.798_{\pm 0.02}$(17)	$0.875_{\pm 0.003}$(5)	$0.521_{\pm 0.057}$(42)	$0.694_{\pm 0.014}$(29)	$0.809_{\pm 0.002}$(14)	$0.756_{\pm 0.003}$(25)
landsat	$0.68_{\pm 0.024}$(9)	$0.458_{\pm 0.003}$(38)	$0.367_{\pm 0.002}$(45)	$0.751_{\pm 0.007}$(3)	$0.495_{\pm 0.003}$(30)	$0.697_{\pm 0.007}$(7)	$0.611_{\pm 0.007}$(14)	$0.739_{\pm 0.003}$(5)	$0.397_{\pm 0.033}$(44)	$0.424_{\pm 0.034}$(43)	$0.613_{\pm 0.002}$(13)	$0.461_{\pm 0.003}$(36)
satimage-2	$0.998_{\pm 0.0}$(6)	$0.922_{\pm 0.002}$(36)	$0.965_{\pm 0.001}$(32)	$0.997_{\pm 0.0}$(10)	$0.995_{\pm 0.001}$(16)	$0.977_{\pm 0.002}$(27)	$0.994_{\pm 0.001}$(19)	$0.999_{\pm 0.0}$(1)	$0.553_{\pm 0.029}$(44)	$0.986_{\pm 0.006}$(24)	$0.996_{\pm 0.0}$(14)	$0.997_{\pm 0.0}$(10)
PageBlocks	$0.954_{\pm 0.008}$(10)	$0.876_{\pm 0.005}$(29)	$0.914_{\pm 0.004}$(25)	$0.971_{\pm 0.001}$(1)	$0.959_{\pm 0.004}$(8)	$0.772_{\pm 0.008}$(37)	$0.929_{\pm 0.006}$(21)	$0.95_{\pm 0.001}$(12)	$0.731_{\pm 0.08}$(41)	$0.871_{\pm 0.049}$(32)	$0.922_{\pm 0.003}$(24)	$0.944_{\pm 0.003}$(16)
Wilt	$0.416_{\pm 0.021}$(35)	$0.619_{\pm 0.019}$(13)	$0.389_{\pm 0.008}$(37)	$0.742_{\pm 0.086}$(8)	$0.722_{\pm 0.021}$(9)	$0.346_{\pm 0.018}$(41)	$0.467_{\pm 0.015}$(28)	$0.346_{\pm 0.006}$(41)	$0.413_{\pm 0.085}$(36)	$0.329_{\pm 0.096}$(44)	$0.859_{\pm 0.004}$(5)	$0.322_{\pm 0.006}$(46)
thyroid	$0.984_{\pm 0.002}$(10)	$0.903_{\pm 0.018}$(36)	$0.978_{\pm 0.002}$(19)	$0.958_{\pm 0.012}$(25)	$0.977_{\pm 0.003}$(21)	$0.981_{\pm 0.002}$(15)	$0.99_{\pm 0.001}$(2)	$0.985_{\pm 0.001}$(8)	$0.959_{\pm 0.009}$(24)	$0.942_{\pm 0.014}$(31)	$0.986_{\pm 0.001}$(7)	$0.984_{\pm 0.001}$(10)
Waveform	$0.726_{\pm 0.017}$(9)	$0.552_{\pm 0.009}$(38)	$0.607_{\pm 0.005}$(32)	$0.778_{\pm 0.005}$(2)	$0.583_{\pm 0.005}$(36)	$0.705_{\pm 0.005}$(13)	$0.723_{\pm 0.023}$(10)	$0.772_{\pm 0.004}$(3)	$0.592_{\pm 0.025}$(33)	$0.618_{\pm 0.055}$(30)	$0.583_{\pm 0.006}$(36)	$0.702_{\pm 0.004}$(15)
Cardiotocography	$0.718_{\pm 0.018}$(23)	$0.621_{\pm 0.012}$(32)	$0.785_{\pm 0.005}$(9)	$0.787_{\pm 0.007}$(8)	$0.759_{\pm 0.004}$(13)	$0.699_{\pm 0.015}$(25)	$0.801_{\pm 0.018}$(5)	$0.771_{\pm 0.002}$(11)	$0.713_{\pm 0.044}$(24)	$0.757_{\pm 0.083}$(14)	$0.676_{\pm 0.02}$(29)	$0.821_{\pm 0.006}$(3)
fault	$0.741_{\pm 0.016}$(7)	$0.623_{\pm 0.03}$(24)	$0.465_{\pm 0.005}$(45)	$0.651_{\pm 0.027}$(20)	$0.69_{\pm 0.011}$(15)	$0.569_{\pm 0.071}$(31)	$0.651_{\pm 0.016}$(20)	$0.815_{\pm 0.008}$(1)	$0.512_{\pm 0.014}$(38)	$0.51_{\pm 0.059}$(40)	$0.63_{\pm 0.03}$(23)	$0.604_{\pm 0.006}$(30)
cardio	$0.959_{\pm 0.016}$(7)	$0.856_{\pm 0.013}$(29)	$0.937_{\pm 0.001}$(15)	$0.92_{\pm 0.017}$(19)	$0.946_{\pm 0.002}$(11)	$0.844_{\pm 0.009}$(32)	$0.94_{\pm 0.01}$(13)	$0.956_{\pm 0.004}$(5)	$0.865_{\pm 0.081}$(28)	$0.883_{\pm 0.036}$(24)	$0.87_{\pm 0.017}$(27)	$0.965_{\pm 0.004}$(2)
letter	$0.779_{\pm 0.014}$(20)	$0.761_{\pm 0.011}$(22)	$0.572_{\pm 0.005}$(36)	$0.87_{\pm 0.006}$(8)	$0.842_{\pm 0.006}$(16)	$0.599_{\pm 0.008}$(35)	$0.621_{\pm 0.015}$(30)	$0.919_{\pm 0.007}$(3)	$0.519_{\pm 0.018}$(39)	$0.541_{\pm 0.041}$(37)	$0.812_{\pm 0.017}$(18)	$0.615_{\pm 0.007}$(31)
yeast	$0.503_{\pm 0.024}$(4)	$0.413_{\pm 0.016}$(39)	$0.446_{\pm 0.015}$(29)	$0.475_{\pm 0.017}$(16)	$0.445_{\pm 0.018}$(30)	$0.417_{\pm 0.015}$(38)	$0.421_{\pm 0.018}$(37)	$0.424_{\pm 0.017}$(34)	$0.481_{\pm 0.015}$(14)	$0.467_{\pm 0.05}$(20)	$0.461_{\pm 0.016}$(22)	$0.454_{\pm 0.016}$(27)
vowels	$0.901_{\pm 0.009}$(17)	$0.878_{\pm 0.005}$(20)	$0.59_{\pm 0.009}$(37)	$0.953_{\pm 0.011}$(12)	$0.947_{\pm 0.006}$(14)	$0.698_{\pm 0.004}$(34)	$0.772_{\pm 0.026}$(27)	$0.963_{\pm 0.005}$(11)	$0.539_{\pm 0.021}$(39)	$0.727_{\pm 0.036}$(30)	$0.739_{\pm 0.08}$(28)	$0.823_{\pm 0.004}$(25)
Avg Ranking	12.87	31.8	30.0	10.07	19.27	26.47	17.67	10.4	33.67	30.2	19.8	20.53

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
annthyroid	$0.759_{\pm 0.004}$(32)	$0.876_{\pm 0.011}$(16)	$0.824_{\pm 0.003}$(26)	$0.517_{\pm 0.034}$(45)	$0.855_{\pm 0.016}$(24)	$0.875_{\pm 0.005}$(18)	$0.16_{\pm 0.016}$(46)	$0.539_{\pm 0.043}$(44)	$0.645_{\pm 0.002}$(40)
pendigits	$0.909_{\pm 0.003}$(25)	$0.959_{\pm 0.012}$(16)	$0.575_{\pm 0.005}$(40)	$0.52_{\pm 0.004}$(44)	$0.854_{\pm 0.01}$(28)	$0.64_{\pm 0.005}$(38)	$0.54_{\pm 0.182}$(43)	$0.497_{\pm 0.004}$(45)	$0.929_{\pm 0.003}$(22)
satellite	$0.859_{\pm 0.004}$(7)	$0.819_{\pm 0.031}$(11)	$0.503_{\pm 0.004}$(46)	$0.516_{\pm 0.006}$(43)	$0.669_{\pm 0.009}$(34)	$0.629_{\pm 0.004}$(37)	$0.512_{\pm 0.05}$(44)	$0.527_{\pm 0.005}$(41)	$0.577_{\pm 0.003}$(39)
landsat	$0.683_{\pm 0.001}$(8)	$0.63_{\pm 0.064}$(10)	$0.456_{\pm 0.002}$(39)	$0.52_{\pm 0.005}$(28)	$0.467_{\pm 0.022}$(35)	$0.53_{\pm 0.004}$(24)	$0.478_{\pm 0.032}$(33)	$0.521_{\pm 0.004}$(26)	$0.343_{\pm 0.001}$(46)
satimage-2	$0.995_{\pm 0.0}$(16)	$0.999_{\pm 0.0}$(1)	$0.727_{\pm 0.002}$(41)	$0.56_{\pm 0.005}$(43)	$0.996_{\pm 0.001}$(14)	$0.906_{\pm 0.002}$(37)	$0.601_{\pm 0.148}$(42)	$0.534_{\pm 0.003}$(45)	$0.976_{\pm 0.001}$(28)
PageBlocks	$0.77_{\pm 0.03}$(38)	$0.944_{\pm 0.006}$(16)	$0.819_{\pm 0.008}$(34)	$0.579_{\pm 0.009}$(45)	$0.948_{\pm 0.007}$(14)	$0.891_{\pm 0.004}$(28)	$0.187_{\pm 0.011}$(46)	$0.702_{\pm 0.008}$(42)	$0.897_{\pm 0.003}$(27)
Wilt	$0.345_{\pm 0.005}$(43)	$0.43_{\pm 0.026}$(32)	$0.545_{\pm 0.006}$(20)	$0.603_{\pm 0.008}$(14)	$0.323_{\pm 0.013}$(45)	$0.503_{\pm 0.004}$(24)	$0.593_{\pm 0.008}$(16)	$0.54_{\pm 0.014}$(21)	$0.429_{\pm 0.005}$(33)
thyroid	$0.827_{\pm 0.029}$(39)	$0.98_{\pm 0.005}$(16)	$0.915_{\pm 0.005}$(35)	$0.435_{\pm 0.015}$(45)	$0.97_{\pm 0.007}$(22)	$0.97_{\pm 0.003}$(22)	$0.166_{\pm 0.047}$(46)	$0.565_{\pm 0.01}$(43)	$0.946_{\pm 0.002}$(27)
Waveform	$0.747_{\pm 0.005}$(7)	$0.7_{\pm 0.058}$(16)	$0.612_{\pm 0.021}$(31)	$0.65_{\pm 0.022}$(26)	$0.722_{\pm 0.012}$(11)	$0.704_{\pm 0.007}$(14)	$0.514_{\pm 0.045}$(40)	$0.671_{\pm 0.012}$(24)	$0.63_{\pm 0.007}$(27)
Cardiotocography	$0.575_{\pm 0.016}$(35)	$0.729_{\pm 0.08}$(22)	$0.408_{\pm 0.006}$(44)	$0.497_{\pm 0.008}$(40)	$0.641_{\pm 0.013}$(30)	$0.462_{\pm 0.004}$(42)	$0.525_{\pm 0.026}$(39)	$0.481_{\pm 0.007}$(41)	$0.695_{\pm 0.008}$(26)
fault	$0.62_{\pm 0.043}$(26)	$0.692_{\pm 0.014}$(14)	$0.656_{\pm 0.006}$(19)	$0.524_{\pm 0.012}$(37)	$0.544_{\pm 0.014}$(34)	$0.677_{\pm 0.008}$(16)	$0.561_{\pm 0.052}$(32)	$0.554_{\pm 0.01}$(33)	$0.471_{\pm 0.027}$(42)
cardio	$0.722_{\pm 0.027}$(37)	$0.928_{\pm 0.026}$(18)	$0.506_{\pm 0.014}$(43)	$0.507_{\pm 0.012}$(42)	$0.885_{\pm 0.011}$(23)	$0.668_{\pm 0.013}$(38)	$0.603_{\pm 0.073}$(39)	$0.506_{\pm 0.011}$(43)	$0.846_{\pm 0.194}$(31)
letter	$0.612_{\pm 0.006}$(34)	$0.729_{\pm 0.022}$(25)	$0.852_{\pm 0.009}$(14)	$0.853_{\pm 0.006}$(13)	$0.696_{\pm 0.009}$(26)	$0.864_{\pm 0.006}$(11)	$0.517_{\pm 0.054}$(40)	$0.841_{\pm 0.002}$(17)	$0.52_{\pm 0.008}$(38)
yeast	$0.406_{\pm 0.013}$(40)	$0.467_{\pm 0.037}$(20)	$0.397_{\pm 0.014}$(42)	$0.441_{\pm 0.02}$(31)	$0.382_{\pm 0.015}$(45)	$0.383_{\pm 0.016}$(44)	$0.494_{\pm 0.016}$(10)	$0.434_{\pm 0.02}$(32)	$0.392_{\pm 0.016}$(43)
vowels	$0.709_{\pm 0.005}$(31)	$0.867_{\pm 0.037}$(21)	$0.91_{\pm 0.003}$(16)	$0.879_{\pm 0.014}$(19)	$0.864_{\pm 0.014}$(22)	$0.952_{\pm 0.007}$(13)	$0.461_{\pm 0.066}$(43)	$0.892_{\pm 0.012}$(18)	$0.591_{\pm 0.016}$(36)
Avg Ranking	27.87	16.93	32.67	34.33	27.13	27.07	37.27	34.33	33.67

Table 9: AUPRC results on 15 medium datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank in terms of mean among all anomaly detectors).

Dataset	TCMM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
annthyroid	$0.695_{\pm 0.065}$(6)	$0.624_{\pm 0.045}$(12)	$0.592_{\pm 0.012}$(17)	$0.202_{\pm 0.028}$(43)	$0.5_{\pm 0.069}$(26)	$0.262_{\pm 0.112}$(40)	$0.588_{\pm 0.065}$(19)	$0.506_{\pm 0.054}$(24)	$0.638_{\pm 0.043}$(9)	$0.861_{\pm 0.008}$(1)	$0.504_{\pm 0.035}$(25)	$0.721_{\pm 0.06}$(2)
pendigits	$0.701_{\pm 0.037}$(8)	$0.565_{\pm 0.127}$(12)	$0.464_{\pm 0.047}$(16)	$0.06_{\pm 0.011}$(44)	$0.172_{\pm 0.137}$(31)	$0.136_{\pm 0.131}$(36)	$0.302_{\pm 0.205}$(24)	$0.726_{\pm 0.074}$(7)	$0.142_{\pm 0.035}$(34)	$0.445_{\pm 0.087}$(18)	$0.209_{\pm 0.031}$(29)	$0.808_{\pm 0.05}$(6)
satellite	$0.861_{\pm 0.004}$(10)	$0.851_{\pm 0.005}$(13)	$0.819_{\pm 0.003}$(24)	$0.493_{\pm 0.029}$(46)	$0.685_{\pm 0.104}$(36)	$0.698_{\pm 0.074}$(34)	$0.764_{\pm 0.054}$(29)	$0.839_{\pm 0.004}$(19)	$0.76_{\pm 0.099}$(30)	$0.853_{\pm 0.005}$(12)	$0.821_{\pm 0.004}$(22)	$0.845_{\pm 0.007}$(17)
landsat	$0.416_{\pm 0.007}$(12)	$0.411_{\pm 0.012}$(14)	$0.413_{\pm 0.01}$(13)	$0.327_{\pm 0.015}$(34)	$0.319_{\pm 0.097}$(38)	$0.365_{\pm 0.057}$(27)	$0.305_{\pm 0.032}$(43)	$0.437_{\pm 0.01}$(11)	$0.411_{\pm 0.014}$(14)	$0.394_{\pm 0.022}$(20)	$0.339_{\pm 0.005}$(32)	$0.342_{\pm 0.016}$(31)
satimage-2	$0.946_{\pm 0.027}$(10)	$0.96_{\pm 0.011}$(9)	$0.899_{\pm 0.005}$(18)	$0.035_{\pm 0.009}$(46)	$0.533_{\pm 0.222}$(33)	$0.124_{\pm 0.078}$(42)	$0.801_{\pm 0.284}$(25)	$0.937_{\pm 0.033}$(14)	$0.241_{\pm 0.194}$(38)	$0.518_{\pm 0.031}$(34)	$0.603_{\pm 0.102}$(32)	$0.815_{\pm 0.039}$(23)
PageBlocks	$0.871_{\pm 0.015}$(2)	$0.828_{\pm 0.04}$(12)	$0.839_{\pm 0.03}$(9)	$0.27_{\pm 0.069}$(45)	$0.583_{\pm 0.033}$(35)	$0.505_{\pm 0.153}$(37)	$0.728_{\pm 0.069}$(23)	$0.788_{\pm 0.016}$(20)	$0.83_{\pm 0.037}$(10)	$0.848_{\pm 0.015}$(7)	$0.679_{\pm 0.029}$(27)	$0.855_{\pm 0.028}$(4)
Wilt	$0.478_{\pm 0.029}$(3)	$0.102_{\pm 0.008}$(20)	$0.09_{\pm 0.004}$(28)	$0.093_{\pm 0.005}$(25)	$0.076_{\pm 0.009}$(38)	$0.147_{\pm 0.047}$(13)	$0.074_{\pm 0.007}$(40)	$0.073_{\pm 0.001}$(41)	$0.093_{\pm 0.019}$(25)	$0.282_{\pm 0.01}$(5)	$0.098_{\pm 0.004}$(22)	$0.132_{\pm 0.008}$(14)
thyroid	$0.802_{\pm 0.07}$(8)	$0.787_{\pm 0.067}$(13)	$0.859_{\pm 0.017}$(2)	$0.106_{\pm 0.027}$(44)	$0.732_{\pm 0.21}$(21)	$0.276_{\pm 0.179}$(41)	$0.814_{\pm 0.037}$(7)	$0.823_{\pm 0.022}$(5)	$0.703_{\pm 0.065}$(22)	$0.873_{\pm 0.016}$(1)	$0.647_{\pm 0.061}$(24)	$0.539_{\pm 0.107}$(31)
Waveform	$0.143_{\pm 0.031}$(12)	$0.118_{\pm 0.014}$(17)	$0.103_{\pm 0.004}$(22)	$0.063_{\pm 0.014}$(42)	$0.089_{\pm 0.009}$(27)	$0.057_{\pm 0.001}$(45)	$0.085_{\pm 0.028}$(29)	$0.125_{\pm 0.009}$(16)	$0.191_{\pm 0.07}$(8)	$0.089_{\pm 0.009}$(27)	$0.071_{\pm 0.015}$(38)	$0.08_{\pm 0.003}$(32)
Cardiotocography	$0.743_{\pm 0.016}$(3)	$0.655_{\pm 0.014}$(16)	$0.661_{\pm 0.066}$(14)	$0.424_{\pm 0.044}$(38)	$0.661_{\pm 0.117}$(14)	*NAN*	$0.718_{\pm 0.101}$(6)	$0.597_{\pm 0.013}$(26)	$0.41_{\pm 0.126}$(41)	$0.624_{\pm 0.018}$(23)	$0.489_{\pm 0.018}$(35)	$0.686_{\pm 0.015}$(11)
fault	$0.769_{\pm 0.035}$(5)	$0.729_{\pm 0.008}$(8)	$0.666_{\pm 0.012}$(17)	$0.529_{\pm 0.032}$(41)	$0.533_{\pm 0.074}$(39)	*NAN*	$0.576_{\pm 0.047}$(33)	$0.717_{\pm 0.021}$(10)	$0.641_{\pm 0.029}$(20)	$0.697_{\pm 0.015}$(14)	$0.629_{\pm 0.015}$(24)	$0.724_{\pm 0.01}$(9)
cardio	$0.847_{\pm 0.022}$(4)	$0.763_{\pm 0.043}$(17)	$0.824_{\pm 0.035}$(9)	$0.233_{\pm 0.03}$(43)	$0.761_{\pm 0.07}$(18)	*NAN*	$0.778_{\pm 0.081}$(13)	$0.836_{\pm 0.021}$(6)	$0.369_{\pm 0.21}$(40)	$0.657_{\pm 0.043}$(27)	$0.473_{\pm 0.023}$(37)	$0.77_{\pm 0.007}$(15)
letter	$0.573_{\pm 0.03}$(5)	$0.358_{\pm 0.031}$(18)	$0.203_{\pm 0.01}$(31)	$0.135_{\pm 0.02}$(41)	$0.124_{\pm 0.008}$(43)	*NAN*	$0.138_{\pm 0.018}$(38)	$0.234_{\pm 0.021}$(28)	$0.247_{\pm 0.018}$(26)	$0.557_{\pm 0.041}$(7)	$0.229_{\pm 0.031}$(29)	$0.553_{\pm 0.041}$(8)
yeast	$0.518_{\pm 0.011}$(3)	$0.485_{\pm 0.014}$(28)	$0.485_{\pm 0.016}$(28)	$0.498_{\pm 0.008}$(15)	$0.472_{\pm 0.032}$(35)	*NAN*	$0.465_{\pm 0.018}$(38)	$0.457_{\pm 0.012}$(40)	$0.508_{\pm 0.037}$(6)	$0.499_{\pm 0.022}$(13)	$0.515_{\pm 0.012}$(5)	$0.499_{\pm 0.02}$(13)
vowels	$0.76_{\pm 0.021}$(8)	$0.543_{\pm 0.03}$(14)	$0.209_{\pm 0.052}$(29)	$0.079_{\pm 0.032}$(41)	$0.069_{\pm 0.011}$(42)	*NAN*	$0.111_{\pm 0.044}$(39)	$0.276_{\pm 0.04}$(26)	$0.111_{\pm 0.063}$(39)	$0.809_{\pm 0.032}$(3)	$0.191_{\pm 0.045}$(31)	$0.771_{\pm 0.028}$(7)
Avg Ranking	6.6	14.87	18.47	39.2	31.73	35.0	27.07	19.53	24.13	14.13	27.47	14.87

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
annthyroid	$0.581_{\pm 0.067}$(20)	$0.694_{\pm 0.012}$(7)	$0.331_{\pm 0.104}$(38)	$0.425_{\pm 0.034}$(31)	$0.444_{\pm 0.015}$(30)	$0.58_{\pm 0.054}$(21)	$0.634_{\pm 0.081}$(11)	$0.355_{\pm 0.041}$(37)	$0.707_{\pm 0.041}$(4)	$0.714_{\pm 0.01}$(3)	$0.3_{\pm 0.026}$(39)	$0.612_{\pm 0.033}$(13)
pendigits	$0.647_{\pm 0.174}$(9)	$0.976_{\pm 0.016}$(2)	$0.094_{\pm 0.057}$(38)	$0.036_{\pm 0.024}$(46)	$0.643_{\pm 0.138}$(10)	$0.983_{\pm 0.011}$(1)	$0.561_{\pm 0.154}$(13)	$0.232_{\pm 0.077}$(28)	$0.137_{\pm 0.061}$(35)	$0.269_{\pm 0.042}$(27)	$0.185_{\pm 0.076}$(30)	$0.419_{\pm 0.052}$(20)
satellite	$0.807_{\pm 0.03}$(26)	$0.893_{\pm 0.003}$(3)	$0.831_{\pm 0.011}$(20)	$0.782_{\pm 0.013}$(28)	$0.899_{\pm 0.006}$(1)	$0.896_{\pm 0.002}$(2)	$0.82_{\pm 0.04}$(23)	$0.748_{\pm 0.01}$(31)	$0.739_{\pm 0.052}$(32)	$0.884_{\pm 0.006}$(5)	$0.722_{\pm 0.025}$(33)	$0.817_{\pm 0.001}$(25)
landsat	$0.37_{\pm 0.015}$(25)	$0.614_{\pm 0.008}$(4)	$0.402_{\pm 0.051}$(18)	$0.377_{\pm 0.005}$(24)	$0.687_{\pm 0.016}$(2)	$0.657_{\pm 0.013}$(3)	$0.394_{\pm 0.04}$(20)	$0.344_{\pm 0.035}$(30)	$0.319_{\pm 0.007}$(38)	$0.495_{\pm 0.006}$(7)	$0.302_{\pm 0.013}$(44)	$0.382_{\pm 0.004}$(22)
satimage-2	$0.647_{\pm 0.148}$(31)	$0.979_{\pm 0.005}$(1)	$0.869_{\pm 0.045}$(21)	$0.971_{\pm 0.005}$(7)	$0.941_{\pm 0.042}$(12)	$0.973_{\pm 0.005}$(5)	$0.75_{\pm 0.299}$(28)	$0.517_{\pm 0.223}$(35)	$0.655_{\pm 0.152}$(30)	$0.927_{\pm 0.035}$(17)	$0.147_{\pm 0.038}$(39)	$0.877_{\pm 0.006}$(20)
PageBlocks	$0.811_{\pm 0.064}$(16)	$0.859_{\pm 0.009}$(3)	$0.472_{\pm 0.167}$(39)	$0.642_{\pm 0.034}$(31)	$0.821_{\pm 0.019}$(15)	$0.83_{\pm 0.015}$(10)	$0.853_{\pm 0.018}$(5)	$0.444_{\pm 0.017}$(40)	$0.687_{\pm 0.071}$(26)	$0.691_{\pm 0.028}$(25)	$0.585_{\pm 0.09}$(34)	$0.794_{\pm 0.016}$(18)
Wilt	$0.745_{\pm 0.06}$(1)	$0.132_{\pm 0.005}$(14)	$0.089_{\pm 0.015}$(29)	$0.17_{\pm 0.018}$(9)	$0.298_{\pm 0.075}$(4)	$0.095_{\pm 0.01}$(24)	$0.16_{\pm 0.16}$(11)	$0.097_{\pm 0.013}$(23)	$0.185_{\pm 0.045}$(8)	$0.158_{\pm 0.003}$(12)	$0.088_{\pm 0.014}$(31)	$0.086_{\pm 0.001}$(32)
thyroid	$0.477_{\pm 0.158}$(34)	$0.797_{\pm 0.03}$(9)	$0.567_{\pm 0.313}$(28)	$0.443_{\pm 0.061}$(36)	$0.477_{\pm 0.118}$(34)	$0.755_{\pm 0.037}$(19)	$0.76_{\pm 0.071}$(18)	$0.37_{\pm 0.119}$(39)	$0.788_{\pm 0.071}$(12)	$0.63_{\pm 0.096}$(26)	$0.482_{\pm 0.087}$(33)	$0.841_{\pm 0.017}$(3)
Waveform	$0.139_{\pm 0.02}$(13)	$0.283_{\pm 0.017}$(5)	$0.132_{\pm 0.046}$(14)	$0.069_{\pm 0.038}$(40)	$0.164_{\pm 0.033}$(10)	$0.306_{\pm 0.016}$(3)	$0.393_{\pm 0.229}$(1)	$0.062_{\pm 0.007}$(43)	$0.223_{\pm 0.045}$(7)	$0.051_{\pm 0.001}$(46)	$0.058_{\pm 0.007}$(44)	$0.1_{\pm 0.008}$(24)
Cardiotocography	$0.647_{\pm 0.058}$(19)	$0.687_{\pm 0.007}$(10)	$0.567_{\pm 0.062}$(28)	$0.258_{\pm 0.02}$(45)	$0.632_{\pm 0.02}$(22)	$0.744_{\pm 0.016}$(2)	$0.703_{\pm 0.045}$(8)	$0.521_{\pm 0.069}$(32)	$0.601_{\pm 0.037}$(25)	$0.555_{\pm 0.018}$(30)	$0.478_{\pm 0.085}$(36)	$0.755_{\pm 0.006}$(1)
fault	$0.702_{\pm 0.017}$(13)	$0.792_{\pm 0.012}$(4)	$0.61_{\pm 0.064}$(29)	$0.64_{\pm 0.057}$(21)	$0.767_{\pm 0.009}$(6)	$0.796_{\pm 0.016}$(2)	$0.707_{\pm 0.016}$(12)	$0.537_{\pm 0.061}$(36)	$0.54_{\pm 0.024}$(34)	$0.793_{\pm 0.009}$(3)	$0.535_{\pm 0.066}$(38)	$0.629_{\pm 0.022}$(24)
cardio	$0.603_{\pm 0.066}$(29)	$0.814_{\pm 0.02}$(11)	$0.687_{\pm 0.054}$(21)	$0.12_{\pm 0.014}$(45)	$0.686_{\pm 0.051}$(23)	$0.853_{\pm 0.029}$(2)	$0.827_{\pm 0.023}$(8)	$0.568_{\pm 0.14}$(31)	$0.735_{\pm 0.022}$(19)	$0.677_{\pm 0.034}$(24)	$0.52_{\pm 0.094}$(33)	$0.875_{\pm 0.016}$(1)
letter	$0.572_{\pm 0.026}$(6)	$0.525_{\pm 0.031}$(11)	$0.31_{\pm 0.09}$(22)	$0.323_{\pm 0.032}$(19)	$0.639_{\pm 0.032}$(1)	$0.627_{\pm 0.034}$(2)	$0.54_{\pm 0.079}$(9)	$0.124_{\pm 0.016}$(43)	$0.315_{\pm 0.067}$(20)	$0.601_{\pm 0.024}$(3)	$0.123_{\pm 0.016}$(45)	$0.19_{\pm 0.012}$(32)
yeast	$0.537_{\pm 0.029}$(2)	$0.494_{\pm 0.018}$(18)	$0.488_{\pm 0.027}$(25)	$0.589_{\pm 0.017}$(1)	$0.506_{\pm 0.013}$(8)	$0.493_{\pm 0.015}$(19)	$0.463_{\pm 0.016}$(39)	$0.49_{\pm 0.024}$(22)	$0.486_{\pm 0.02}$(27)	$0.516_{\pm 0.009}$(4)	$0.5_{\pm 0.024}$(12)	$0.482_{\pm 0.011}$(30)
vowels	$0.874_{\pm 0.039}$(2)	$0.796_{\pm 0.064}$(5)	$0.199_{\pm 0.121}$(30)	$0.322_{\pm 0.08}$(23)	$0.885_{\pm 0.03}$(1)	$0.809_{\pm 0.062}$(3)	$0.688_{\pm 0.032}$(12)	$0.038_{\pm 0.004}$(45)	$0.39_{\pm 0.102}$(17)	$0.758_{\pm 0.023}$(9)	$0.039_{\pm 0.007}$(44)	$0.128_{\pm 0.029}$(36)
Avg Ranking	16.4	7.13	26.67	27.07	11.93	7.87	14.53	34.33	22.27	16.07	35.67	20.07

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
annthyroid	$0.636_{\pm 0.057}$(10)	$0.226_{\pm 0.011}$(42)	$0.402_{\pm 0.006}$(34)	$0.54_{\pm 0.122}$(23)	$0.543_{\pm 0.039}$(22)	$0.419_{\pm 0.047}$(32)	$0.612_{\pm 0.044}$(13)	$0.601_{\pm 0.041}$(15)	$0.46_{\pm 0.05}$(28)	$0.404_{\pm 0.045}$(33)	$0.645_{\pm 0.01}$(8)	$0.592_{\pm 0.042}$(17)
pendigits	$0.454_{\pm 0.105}$(17)	$0.051_{\pm 0.001}$(45)	$0.417_{\pm 0.009}$(21)	$0.93_{\pm 0.035}$(4)	$0.156_{\pm 0.004}$(33)	$0.426_{\pm 0.012}$(19)	$0.57_{\pm 0.054}$(11)	$0.969_{\pm 0.016}$(3)	$0.279_{\pm 0.09}$(26)	$0.394_{\pm 0.075}$(22)	$0.132_{\pm 0.006}$(37)	$0.523_{\pm 0.033}$(14)
satellite	$0.87_{\pm 0.015}$(9)	$0.582_{\pm 0.005}$(40)	$0.66_{\pm 0.004}$(38)	$0.879_{\pm 0.003}$(7)	$0.848_{\pm 0.002}$(15)	$0.883_{\pm 0.004}$(6)	$0.843_{\pm 0.012}$(18)	$0.891_{\pm 0.003}$(4)	$0.555_{\pm 0.055}$(41)	$0.791_{\pm 0.013}$(27)	$0.849_{\pm 0.003}$(14)	$0.823_{\pm 0.002}$(21)
landsat	$0.467_{\pm 0.026}$(9)	$0.314_{\pm 0.002}$(41)	$0.281_{\pm 0.001}$(45)	$0.693_{\pm 0.006}$(1)	$0.337_{\pm 0.002}$(33)	$0.526_{\pm 0.01}$(6)	$0.445_{\pm 0.025}$(10)	$0.556_{\pm 0.005}$(5)	$0.379_{\pm 0.075}$(23)	$0.326_{\pm 0.043}$(35)	$0.401_{\pm 0.002}$(19)	$0.325_{\pm 0.002}$(36)
satimage-2	$0.976_{\pm 0.002}$(4)	$0.268_{\pm 0.014}$(37)	$0.746_{\pm 0.015}$(29)	$0.94_{\pm 0.009}$(13)	$0.798_{\pm 0.04}$(26)	$0.833_{\pm 0.009}$(22)	$0.932_{\pm 0.005}$(15)	$0.979_{\pm 0.004}$(1)	$0.036_{\pm 0.013}$(45)	$0.931_{\pm 0.015}$(16)	$0.815_{\pm 0.008}$(23)	$0.973_{\pm 0.002}$(5)
PageBlocks	$0.825_{\pm 0.038}$(13)	$0.545_{\pm 0.011}$(36)	$0.659_{\pm 0.014}$(30)	$0.888_{\pm 0.007}$(1)	$0.822_{\pm 0.02}$(14)	$0.353_{\pm 0.011}$(41)	$0.699_{\pm 0.026}$(24)	$0.844_{\pm 0.009}$(8)	$0.502_{\pm 0.092}$(38)	$0.664_{\pm 0.071}$(29)	$0.74_{\pm 0.013}$(22)	$0.796_{\pm 0.013}$(17)
Wilt	$0.079_{\pm 0.003}$(36)	$0.121_{\pm 0.008}$(18)	$0.079_{\pm 0.001}$(36)	$0.19_{\pm 0.053}$(7)	$0.163_{\pm 0.012}$(10)	$0.076_{\pm 0.002}$(38)	$0.086_{\pm 0.002}$(32)	$0.071_{\pm 0.001}$(42)	$0.089_{\pm 0.009}$(29)	$0.071_{\pm 0.01}$(42)	$0.261_{\pm 0.005}$(6)	$0.069_{\pm 0.001}$(45)
thyroid	$0.787_{\pm 0.027}$(13)	$0.299_{\pm 0.042}$(40)	$0.636_{\pm 0.023}$(25)	$0.426_{\pm 0.16}$(38)	$0.761_{\pm 0.011}$(17)	$0.747_{\pm 0.023}$(20)	$0.821_{\pm 0.033}$(6)	$0.791_{\pm 0.011}$(10)	$0.697_{\pm 0.047}$(23)	$0.597_{\pm 0.09}$(27)	$0.825_{\pm 0.016}$(4)	$0.79_{\pm 0.03}$(11)
Waveform	$0.246_{\pm 0.008}$(6)	$0.071_{\pm 0.001}$(38)	$0.078_{\pm 0.003}$(33)	$0.318_{\pm 0.036}$(2)	$0.078_{\pm 0.002}$(33)	$0.094_{\pm 0.003}$(26)	$0.111_{\pm 0.008}$(20)	$0.289_{\pm 0.015}$(4)	$0.072_{\pm 0.009}$(37)	$0.078_{\pm 0.009}$(33)	$0.078_{\pm 0.002}$(33)	$0.112_{\pm 0.004}$(19)
Cardiotocography	$0.644_{\pm 0.018}$(20)	$0.511_{\pm 0.011}$(33)	$0.654_{\pm 0.006}$(17)	$0.683_{\pm 0.012}$(12)	$0.676_{\pm 0.007}$(13)	$0.584_{\pm 0.011}$(27)	$0.689_{\pm 0.021}$(9)	$0.709_{\pm 0.007}$(7)	$0.605_{\pm 0.053}$(24)	$0.639_{\pm 0.09}$(21)	$0.531_{\pm 0.029}$(31)	$0.731_{\pm 0.01}$(5)
fault	$0.717_{\pm 0.022}$(10)	$0.609_{\pm 0.033}$(30)	$0.488_{\pm 0.006}$(45)	$0.616_{\pm 0.027}$(26)	$0.68_{\pm 0.016}$(15)	$0.579_{\pm 0.062}$(32)	$0.646_{\pm 0.02}$(19)	$0.806_{\pm 0.013}$(1)	$0.51_{\pm 0.01}$(43)	$0.525_{\pm 0.056}$(42)	$0.612_{\pm 0.029}$(27)	$0.64_{\pm 0.015}$(21)
cardio	$0.83_{\pm 0.039}$(7)	$0.52_{\pm 0.028}$(33)	$0.717_{\pm 0.007}$(20)	$0.676_{\pm 0.05}$(25)	$0.811_{\pm 0.01}$(12)	$0.612_{\pm 0.015}$(28)	$0.769_{\pm 0.032}$(16)	$0.846_{\pm 0.015}$(5)	$0.687_{\pm 0.114}$(21)	$0.668_{\pm 0.111}$(26)	$0.549_{\pm 0.041}$(32)	$0.85_{\pm 0.01}$(3)
letter	$0.296_{\pm 0.024}$(24)	$0.299_{\pm 0.01}$(23)	$0.142_{\pm 0.003}$(36)	$0.503_{\pm 0.038}$(12)	$0.433_{\pm 0.023}$(15)	$0.15_{\pm 0.005}$(35)	$0.159_{\pm 0.007}$(34)	$0.588_{\pm 0.03}$(4)	$0.13_{\pm 0.006}$(42)	$0.138_{\pm 0.015}$(38)	$0.313_{\pm 0.032}$(21)	$0.207_{\pm 0.01}$(30)
yeast	$0.508_{\pm 0.02}$(6)	$0.452_{\pm 0.01}$(42)	$0.498_{\pm 0.013}$(15)	$0.503_{\pm 0.019}$(11)	$0.477_{\pm 0.014}$(33)	$0.492_{\pm 0.012}$(20)	$0.477_{\pm 0.013}$(33)	$0.48_{\pm 0.013}$(32)	$0.506_{\pm 0.009}$(8)	$0.495_{\pm 0.031}$(17)	$0.488_{\pm 0.013}$(25)	$0.489_{\pm 0.013}$(24)
vowels	$0.332_{\pm 0.032}$(22)	$0.333_{\pm 0.013}$(21)	$0.146_{\pm 0.011}$(35)	$0.622_{\pm 0.067}$(13)	$0.689_{\pm 0.068}$(11)	$0.181_{\pm 0.022}$(33)	$0.232_{\pm 0.053}$(27)	$0.74_{\pm 0.066}$(10)	$0.121_{\pm 0.027}$(37)	$0.224_{\pm 0.058}$(28)	$0.153_{\pm 0.097}$(34)	$0.377_{\pm 0.026}$(18)
Avg Ranking	13.73	34.6	30.6	13.0	20.13	25.67	19.13	10.07	31.0	29.07	22.4	19.07

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
annthyroid	$0.373_{\pm 0.011}$(35)	$0.598_{\pm 0.022}$(16)	$0.356_{\pm 0.007}$(36)	$0.143_{\pm 0.012}$(45)	$0.456_{\pm 0.042}$(29)	$0.488_{\pm 0.016}$(27)	$0.082_{\pm 0.002}$(46)	$0.178_{\pm 0.017}$(44)	$0.262_{\pm 0.005}$(40)
pendigits	$0.282_{\pm 0.014}$(25)	$0.499_{\pm 0.089}$(15)	$0.07_{\pm 0.005}$(40)	$0.063_{\pm 0.004}$(42)	$0.17_{\pm 0.022}$(32)	$0.088_{\pm 0.003}$(39)	$0.064_{\pm 0.034}$(41)	$0.061_{\pm 0.003}$(43)	$0.337_{\pm 0.006}$(23)
satellite	$0.877_{\pm 0.005}$(8)	$0.848_{\pm 0.024}$(15)	$0.503_{\pm 0.004}$(45)	$0.52_{\pm 0.004}$(43)	$0.688_{\pm 0.015}$(35)	$0.637_{\pm 0.002}$(39)	$0.529_{\pm 0.041}$(42)	$0.508_{\pm 0.002}$(44)	$0.674_{\pm 0.003}$(37)
landsat	$0.476_{\pm 0.007}$(8)	$0.405_{\pm 0.058}$(17)	$0.316_{\pm 0.002}$(40)	$0.366_{\pm 0.005}$(26)	$0.324_{\pm 0.014}$(37)	$0.359_{\pm 0.002}$(29)	$0.309_{\pm 0.014}$(42)	$0.36_{\pm 0.005}$(28)	$0.266_{\pm 0.0}$(46)
satimage-2	$0.942_{\pm 0.004}$(11)	$0.978_{\pm 0.007}$(3)	$0.127_{\pm 0.007}$(41)	$0.093_{\pm 0.029}$(43)	$0.798_{\pm 0.066}$(26)	$0.37_{\pm 0.025}$(36)	$0.134_{\pm 0.024}$(40)	$0.051_{\pm 0.01}$(44)	$0.887_{\pm 0.004}$(19)
PageBlocks	$0.347_{\pm 0.023}$(43)	$0.793_{\pm 0.018}$(19)	$0.588_{\pm 0.01}$(33)	$0.29_{\pm 0.006}$(44)	$0.779_{\pm 0.019}$(21)	$0.677_{\pm 0.008}$(28)	$0.119_{\pm 0.002}$(46)	$0.348_{\pm 0.008}$(42)	$0.633_{\pm 0.01}$(32)
Wilt	$0.071_{\pm 0.001}$(42)	$0.081_{\pm 0.004}$(34)	$0.102_{\pm 0.002}$(20)	$0.124_{\pm 0.002}$(17)	$0.069_{\pm 0.001}$(45)	$0.092_{\pm 0.001}$(27)	$0.127_{\pm 0.006}$(16)	$0.103_{\pm 0.004}$(19)	$0.081_{\pm 0.001}$(34)
thyroid	$0.483_{\pm 0.025}$(32)	$0.784_{\pm 0.061}$(15)	$0.228_{\pm 0.013}$(42)	$0.048_{\pm 0.002}$(45)	$0.564_{\pm 0.096}$(29)	$0.549_{\pm 0.039}$(30)	$0.028_{\pm 0.001}$(46)	$0.11_{\pm 0.008}$(43)	$0.429_{\pm 0.02}$(37)
Waveform	$0.111_{\pm 0.004}$(20)	$0.17_{\pm 0.047}$(9)	$0.083_{\pm 0.009}$(31)	$0.098_{\pm 0.011}$(25)	$0.126_{\pm 0.008}$(15)	$0.145_{\pm 0.013}$(11)	$0.068_{\pm 0.018}$(41)	$0.103_{\pm 0.006}$(22)	$0.084_{\pm 0.003}$(30)
Cardiotocography	$0.429_{\pm 0.013}$(37)	$0.654_{\pm 0.072}$(17)	$0.35_{\pm 0.004}$(44)	$0.387_{\pm 0.015}$(42)	$0.501_{\pm 0.008}$(34)	$0.422_{\pm 0.007}$(39)	$0.412_{\pm 0.021}$(40)	$0.375_{\pm 0.011}$(43)	$0.566_{\pm 0.006}$(29)
fault	$0.611_{\pm 0.035}$(28)	$0.674_{\pm 0.014}$(16)	$0.634_{\pm 0.006}$(23)	$0.53_{\pm 0.007}$(40)	$0.583_{\pm 0.016}$(31)	$0.651_{\pm 0.016}$(18)	$0.539_{\pm 0.039}$(35)	$0.539_{\pm 0.008}$(37)	$0.49_{\pm 0.023}$(44)
cardio	$0.475_{\pm 0.043}$(36)	$0.775_{\pm 0.084}$(14)	$0.241_{\pm 0.015}$(41)	$0.223_{\pm 0.021}$(44)	$0.499_{\pm 0.033}$(35)	$0.414_{\pm 0.02}$(38)	$0.379_{\pm 0.055}$(39)	$0.24_{\pm 0.009}$(42)	$0.585_{\pm 0.229}$(30)
letter	$0.167_{\pm 0.008}$(33)	$0.237_{\pm 0.028}$(27)	$0.398_{\pm 0.016}$(17)	$0.536_{\pm 0.02}$(10)	$0.276_{\pm 0.013}$(25)	$0.419_{\pm 0.015}$(16)	$0.141_{\pm 0.025}$(37)	$0.441_{\pm 0.015}$(14)	$0.136_{\pm 0.003}$(40)
yeast	$0.49_{\pm 0.014}$(22)	$0.491_{\pm 0.018}$(21)	$0.453_{\pm 0.01}$(41)	$0.468_{\pm 0.012}$(37)	$0.447_{\pm 0.01}$(44)	$0.446_{\pm 0.012}$(45)	$0.481_{\pm 0.01}$(31)	$0.47_{\pm 0.014}$(36)	$0.452_{\pm 0.011}$(42)
vowels	$0.186_{\pm 0.022}$(32)	$0.307_{\pm 0.102}$(24)	$0.461_{\pm 0.042}$(16)	$0.335_{\pm 0.036}$(20)	$0.345_{\pm 0.054}$(19)	$0.498_{\pm 0.059}$(15)	$0.064_{\pm 0.012}$(43)	$0.301_{\pm 0.017}$(25)	$0.114_{\pm 0.012}$(38)
Avg Ranking	27.47	17.47	34.0	34.87	30.47	29.13	39.0	35.07	34.73

Table 10: AUROC results on 11 large datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank).

Dataset	TCCM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
ALOI	$0.565_{\pm0.014}$(8)	$0.559_{\pm0.003}$(14)	$0.552_{\pm0.005}$(19)	$0.509_{\pm0.012}$(39)	$0.535_{\pm0.008}$(29)	$0.507_{\pm0.015}$(40)	$0.548_{\pm0.012}$(24)	$0.55_{\pm0.002}$(21)	$0.5_{\pm0.0}$(43)	$0.533_{\pm0.003}$(30)	$0.533_{\pm0.003}$(30)	$0.565_{\pm0.01}$(8)
celeba	$0.76_{\pm0.049}$(14)	$0.73_{\pm0.03}$(18)	$0.764_{\pm0.001}$(12)	$0.511_{\pm0.011}$(37)	$0.697_{\pm0.053}$(22)	$0.585_{\pm0.097}$(34)	$0.736_{\pm0.085}$(17)	$0.653_{\pm0.015}$(30)	$0.5_{\pm0.0}$(38)	$0.801_{\pm0.027}$(2)	$0.664_{\pm0.011}$(26)	$0.656_{\pm0.054}$(29)
cover	$0.983_{\pm0.002}$(5)	$0.978_{\pm0.012}$(9)	$0.984_{\pm0.001}$(4)	$0.521_{\pm0.021}$(40)	$0.759_{\pm0.16}$(30)	$0.813_{\pm0.105}$(29)	$0.874_{\pm0.063}$(24)	$0.978_{\pm0.007}$(9)	$0.5_{\pm0.0}$(42)	$0.979_{\pm0.006}$(8)	$0.703_{\pm0.021}$(33)	$0.98_{\pm0.01}$(7)
donors	$0.998_{\pm0.003}$(6)	$0.93_{\pm0.021}$(11)	$0.452_{\pm0.035}$(38)	$0.547_{\pm0.021}$(34)	$0.64_{\pm0.217}$(31)	$0.708_{\pm0.309}$(29)	$0.836_{\pm0.072}$(20)	$0.89_{\pm0.014}$(16)	$0.5_{\pm0.0}$(37)	$0.973_{\pm0.011}$(10)	$0.814_{\pm0.014}$(22)	$0.999_{\pm0.0}$(3)
fraud	$0.957_{\pm0.003}$(5)	$0.959_{\pm0.001}$(3)	$0.955_{\pm0.001}$(10)	$0.569_{\pm0.008}$(40)	$0.954_{\pm0.004}$(13)	$0.752_{\pm0.085}$(38)	$0.939_{\pm0.009}$(29)	$0.95_{\pm0.002}$(18)	$0.5_{\pm0.0}$(41)	$0.949_{\pm0.007}$(21)	$0.94_{\pm0.002}$(28)	$0.948_{\pm0.008}$(24)
http	$1.0_{\pm0.0}$(1)	$0.999_{\pm0.001}$(6)	$0.862_{\pm0.223}$(31)	$0.634_{\pm0.059}$(35)	$0.999_{\pm0.0}$(6)	$0.988_{\pm0.014}$(27)	$0.998_{\pm0.002}$(16)	$0.993_{\pm0.0}$(24)	$0.5_{\pm0.0}$(36)	$0.995_{\pm0.001}$(21)	$0.982_{\pm0.027}$(28)	$0.761_{\pm0.365}$(32)
magic.gamma	$0.837_{\pm0.009}$(9)	$0.826_{\pm0.007}$(10)	$0.673_{\pm0.008}$(34)	$0.537_{\pm0.016}$(43)	$0.676_{\pm0.063}$(33)	$0.606_{\pm0.068}$(41)	$0.677_{\pm0.033}$(32)	$0.765_{\pm0.004}$(15)	$0.784_{\pm0.011}$(12)	$0.874_{\pm0.002}$(1)	$0.698_{\pm0.007}$(30)	$0.862_{\pm0.007}$(2)
mammography	$0.888_{\pm0.018}$(3)	$0.884_{\pm0.034}$(6)	$0.69_{\pm0.071}$(40)	$0.513_{\pm0.051}$(45)	$0.884_{\pm0.018}$(6)	$0.857_{\pm0.024}$(17)	$0.852_{\pm0.044}$(19)	$0.827_{\pm0.01}$(25)	$0.814_{\pm0.019}$(29)	$0.859_{\pm0.021}$(16)	$0.742_{\pm0.012}$(36)	$0.861_{\pm0.013}$(15)
shuttle	$0.999_{\pm0.0}$(5)	$0.998_{\pm0.001}$(9)	$0.996_{\pm0.001}$(17)	$0.659_{\pm0.04}$(38)	$0.992_{\pm0.002}$(23)	$0.979_{\pm0.009}$(30)	$0.993_{\pm0.001}$(21)	$0.991_{\pm0.001}$(24)	$0.5_{\pm0.0}$(40)	$0.997_{\pm0.0}$(11)	$0.997_{\pm0.0}$(11)	$1.0_{\pm0.0}$(1)
skin	$0.847_{\pm0.094}$(17)	$0.831_{\pm0.074}$(21)	$0.603_{\pm0.004}$(30)	$0.528_{\pm0.03}$(34)	$0.607_{\pm0.111}$(29)	$0.836_{\pm0.077}$(18)	$0.618_{\pm0.118}$(28)	$0.834_{\pm0.004}$(19)	$0.92_{\pm0.003}$(7)	$0.92_{\pm0.002}$(7)	$0.851_{\pm0.006}$(16)	$0.991_{\pm0.001}$(2)
smtp	$0.912_{\pm0.068}$(8)	$0.923_{\pm0.008}$(5)	$0.796_{\pm0.008}$(32)	$0.613_{\pm0.068}$(39)	$0.889_{\pm0.021}$(15)	$0.854_{\pm0.055}$(22)	$0.808_{\pm0.038}$(30)	$0.847_{\pm0.004}$(24)	$0.5_{\pm0.0}$(43)	$0.95_{\pm0.002}$(1)	$0.842_{\pm0.035}$(26)	$0.899_{\pm0.024}$(12)
Avg Ranking	7.36	10.18	24.27	38.55	21.55	29.55	23.64	20.45	33.45	11.64	26.0	12.27

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
ALOI	$0.572_{\pm0.013}$(7)	$0.7_{\pm0.002}$(5)	$0.547_{\pm0.008}$(25)	$0.506_{\pm0.005}$(41)	$0.529_{\pm0.007}$(34)	$0.734_{\pm0.007}$(2)	$0.562_{\pm0.007}$(11)	$0.542_{\pm0.008}$(27)	$0.506_{\pm0.015}$(41)	$0.549_{\pm0.004}$(22)	$0.544_{\pm0.004}$(26)	$0.555_{\pm0.003}$(17)
celeba	$0.775_{\pm0.102}$(9)	$0.663_{\pm0.002}$(27)	$0.368_{\pm0.059}$(42)	NAN	$0.713_{\pm0.018}$(20)	$0.629_{\pm0.004}$(31)	$0.777_{\pm0.021}$(8)	$0.665_{\pm0.062}$(25)	NAN	$0.622_{\pm0.015}$(32)	$0.544_{\pm0.021}$(35)	$0.767_{\pm0.001}$(11)
cover	$0.982_{\pm0.01}$(6)	$0.989_{\pm0.0}$(2)	$0.659_{\pm0.192}$(35)	NAN	$0.9_{\pm0.046}$(22)	$0.993_{\pm0.0}$(1)	$0.826_{\pm0.099}$(27)	$0.633_{\pm0.017}$(36)	NAN	$0.83_{\pm0.01}$(26)	$0.549_{\pm0.016}$(39)	$0.975_{\pm0.0}$(11)
donors	$0.875_{\pm0.127}$(18)	$0.999_{\pm0.0}$(3)	$0.761_{\pm0.232}$(25)	NAN	$1.0_{\pm0.0}$(1)	$1.0_{\pm0.0}$(1)	$0.998_{\pm0.001}$(6)	$0.065_{\pm0.027}$(41)	NAN	NAN	$0.306_{\pm0.146}$(40)	$0.813_{\pm0.047}$(23)
fraud	$0.941_{\pm0.01}$(27)	$0.963_{\pm0.001}$(2)	$0.912_{\pm0.043}$(33)	NAN	$0.937_{\pm0.01}$(30)	$0.965_{\pm0.002}$(1)	$0.957_{\pm0.004}$(5)	$0.775_{\pm0.011}$(37)	NAN	$0.947_{\pm0.004}$(25)	$0.598_{\pm0.019}$(39)	$0.955_{\pm0.001}$(10)
http	$1.0_{\pm0.0}$(1)	$1.0_{\pm0.0}$(1)	$0.493_{\pm0.312}$(37)	NAN	$0.999_{\pm0.001}$(6)	$0.997_{\pm0.003}$(17)	$0.997_{\pm0.003}$(17)	$0.133_{\pm0.18}$(41)	NAN	NAN	$0.736_{\pm0.036}$(33)	$0.999_{\pm0.0}$(6)
magic.gamma	$0.852_{\pm0.022}$(4)	$0.839_{\pm0.001}$(8)	$0.578_{\pm0.014}$(42)	$0.634_{\pm0.008}$(38)	$0.764_{\pm0.004}$(16)	$0.853_{\pm0.004}$(3)	$0.844_{\pm0.038}$(5)	$0.441_{\pm0.049}$(46)	$0.749_{\pm0.014}$(21)	$0.725_{\pm0.004}$(26)	$0.504_{\pm0.034}$(45)	$0.708_{\pm0.02}$(28)
mammography	$0.869_{\pm0.028}$(13)	$0.876_{\pm0.004}$(12)	$0.853_{\pm0.074}$(18)	$0.681_{\pm0.024}$(42)	$0.76_{\pm0.03}$(35)	$0.881_{\pm0.005}$(8)	$0.848_{\pm0.045}$(21)	$0.71_{\pm0.023}$(39)	$0.812_{\pm0.027}$(30)	$0.765_{\pm0.036}$(34)	$0.784_{\pm0.035}$(33)	$0.797_{\pm0.008}$(32)
shuttle	$1.0_{\pm0.0}$(1)	$0.999_{\pm0.0}$(5)	$0.974_{\pm0.009}$(31)	$0.989_{\pm0.004}$(26)	$1.0_{\pm0.0}$(1)	$1.0_{\pm0.0}$(1)	$0.998_{\pm0.001}$(9)	$0.02_{\pm0.006}$(44)	$0.853_{\pm0.07}$(34)	$0.999_{\pm0.0}$(5)	$0.072_{\pm0.1}$(43)	$0.996_{\pm0.0}$(17)
skin	$0.984_{\pm0.006}$(4)	$0.998_{\pm0.0}$(1)	$0.436_{\pm0.032}$(38)	NAN	$0.074_{\pm0.011}$(42)	$0.991_{\pm0.001}$(2)	$0.691_{\pm0.096}$(25)	$0.534_{\pm0.114}$(33)	NAN	$0.929_{\pm0.015}$(5)	$0.383_{\pm0.088}$(40)	$0.628_{\pm0.002}$(26)
smtp	$0.857_{\pm0.037}$(20)	$0.933_{\pm0.008}$(4)	$0.504_{\pm0.094}$(42)	$0.906_{\pm0.015}$(11)	$0.697_{\pm0.11}$(38)	$0.935_{\pm0.012}$(3)	$0.845_{\pm0.044}$(25)	$0.587_{\pm0.028}$(40)	NAN	$0.922_{\pm0.005}$(6)	$0.551_{\pm0.017}$(41)	$0.837_{\pm0.015}$(27)
Avg Ranking	10.0	6.36	33.45	31.6	22.27	6.36	14.45	37.18	31.5	20.11	37.64	18.91

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
ALOI	$0.557_{\pm0.001}$(16)	$0.517_{\pm0.002}$(37)	$0.531_{\pm0.001}$(32)	$0.765_{\pm0.003}$(1)	$0.561_{\pm0.001}$(12)	$0.53_{\pm0.001}$(33)	$0.542_{\pm0.002}$(27)	$0.563_{\pm0.001}$(10)	$0.513_{\pm0.004}$(38)	$0.486_{\pm0.02}$(44)	$0.52_{\pm0.003}$(36)	$0.551_{\pm0.001}$(20)
celeba	$0.781_{\pm0.022}$(6)	$0.708_{\pm0.005}$(21)	$0.757_{\pm0.0}$(15)	$0.534_{\pm0.016}$(36)	$0.807_{\pm0.0}$(1)	$0.761_{\pm0.001}$(13)	$0.718_{\pm0.012}$(19)	$0.675_{\pm0.002}$(24)	$0.688_{\pm NAN}$(23)	$0.661_{\pm0.121}$(28)	$0.799_{\pm0.042}$(3)	$0.79_{\pm0.0}$(4)
cover	$0.943_{\pm0.005}$(17)	$0.744_{\pm0.025}$(31)	$0.92_{\pm0.001}$(20)	$0.989_{\pm0.005}$(2)	$0.95_{\pm0.001}$(16)	$0.72_{\pm0.001}$(32)	$0.843_{\pm0.011}$(25)	$0.956_{\pm0.001}$(14)	$0.893_{\pm0.048}$(23)	$0.94_{\pm0.031}$(18)	$0.702_{\pm0.001}$(34)	$0.963_{\pm0.001}$(13)
donors	$0.929_{\pm0.006}$(12)	$0.504_{\pm0.068}$(36)	$0.889_{\pm0.0}$(17)	$0.99_{\pm0.003}$(8)	$0.925_{\pm0.0}$(13)	$0.797_{\pm0.014}$(24)	$0.891_{\pm0.012}$(15)	$0.975_{\pm0.0}$(9)	$0.725_{\pm NAN}$(27)	$0.646_{\pm0.318}$(30)	$0.822_{\pm0.106}$(21)	$0.921_{\pm0.0}$(14)
fraud	$0.951_{\pm0.006}$(15)	$0.949_{\pm0.003}$(21)	$0.95_{\pm0.0}$(18)	$0.821_{\pm0.042}$(35)	$0.957_{\pm0.0}$(5)	$0.951_{\pm0.001}$(15)	$0.949_{\pm0.002}$(21)	$0.959_{\pm0.001}$(3)	$0.942_{\pm NAN}$(26)	$0.816_{\pm0.168}$(36)	$0.922_{\pm0.003}$(31)	$0.956_{\pm0.0}$(8)
http	$0.999_{\pm0.0}$(6)	$0.994_{\pm0.0}$(23)	$0.979_{\pm0.0}$(29)	$0.897_{\pm0.006}$(30)	$0.999_{\pm0.0}$(6)	$0.993_{\pm0.003}$(24)	$0.992_{\pm0.003}$(26)	$1.0_{\pm0.0}$(1)	$0.999_{\pm0.0}$(6)	$0.179_{\pm0.225}$(40)	$0.999_{\pm0.0}$(6)	$1.0_{\pm0.0}$(1)
magic.gamma	$0.756_{\pm0.01}$(20)	$0.738_{\pm0.001}$(24)	$0.637_{\pm0.001}$(37)	$0.84_{\pm0.003}$(7)	$0.804_{\pm0.001}$(11)	$0.745_{\pm0.004}$(22)	$0.77_{\pm0.012}$(14)	$0.763_{\pm0.001}$(17)	$0.628_{\pm0.007}$(39)	$0.705_{\pm0.016}$(29)	$0.738_{\pm0.004}$(24)	$0.743_{\pm0.001}$(23)
mammography	$0.844_{\pm0.017}$(23)	$0.823_{\pm0.007}$(26)	$0.906_{\pm0.002}$(1)	$0.845_{\pm0.012}$(22)	$0.878_{\pm0.002}$(9)	$0.844_{\pm0.006}$(23)	$0.877_{\pm0.012}$(11)	$0.878_{\pm0.003}$(9)	$0.85_{\pm0.018}$(20)	$0.9_{\pm0.006}$(2)	$0.723_{\pm0.026}$(37)	$0.885_{\pm0.003}$(4)
shuttle	$0.997_{\pm0.001}$(11)	$0.76_{\pm0.02}$(35)	$0.993_{\pm0.0}$(21)	$0.873_{\pm0.085}$(33)	$0.994_{\pm0.001}$(20)	$0.988_{\pm0.002}$(27)	$0.997_{\pm0.001}$(11)	$0.997_{\pm0.0}$(11)	$0.984_{\pm0.007}$(29)	$0.683_{\pm0.365}$(37)	$0.99_{\pm0.001}$(25)	$0.996_{\pm0.0}$(17)
skin	$0.924_{\pm0.012}$(6)	$0.734_{\pm0.002}$(24)	$0.489_{\pm0.001}$(36)	$0.852_{\pm0.014}$(15)	$0.888_{\pm0.001}$(13)	$0.77_{\pm0.001}$(22)	$0.89_{\pm0.005}$(12)	$0.891_{\pm0.001}$(11)	$0.42_{\pm0.06}$(39)	$0.742_{\pm0.057}$(23)	$0.884_{\pm0.001}$(14)	$0.903_{\pm0.001}$(10)
smtp	$0.892_{\pm0.027}$(14)	$0.784_{\pm0.001}$(35)	$0.88_{\pm0.0}$(17)	$0.829_{\pm0.032}$(28)	$0.815_{\pm0.001}$(29)	$0.795_{\pm0.007}$(33)	$0.907_{\pm0.009}$(10)	$0.881_{\pm0.001}$(16)	$0.795_{\pm0.073}$(33)	$0.735_{\pm0.019}$(36)	$0.95_{\pm0.0}$(1)	$0.854_{\pm0.001}$(22)
Avg Ranking	13.27	28.45	22.09	19.73	12.27	24.36	17.36	11.36	27.55	29.36	21.09	12.36

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
ALOI	$0.526_{\pm0.001}$(35)	$0.553_{\pm0.004}$(18)	$0.728_{\pm0.002}$(4)	NAN	$0.558_{\pm0.002}$(15)	$0.671_{\pm0.002}$(6)	NAN	$0.731_{\pm0.002}$(3)	$0.549_{\pm0.001}$(22)
celeba	$0.5_{\pm0.0}$(38)	$0.787_{\pm0.032}$(5)	$0.478_{\pm0.004}$(40)	NAN	$0.755_{\pm0.003}$(16)	$0.589_{\pm0.001}$(33)	NAN	$0.443_{\pm0.005}$(41)	$0.771_{\pm0.0}$(10)
cover	$0.82_{\pm0.001}$(28)	$0.911_{\pm0.041}$(21)	$0.61_{\pm0.0}$(38)	NAN	$0.951_{\pm0.003}$(15)	$0.627_{\pm0.0}$(37)	NAN	$0.503_{\pm0.001}$(41)	$0.923_{\pm0.0}$(19)
donors	$0.724_{\pm0.004}$(28)	$0.857_{\pm0.042}$(19)	$0.441_{\pm0.002}$(39)	NAN	$0.594_{\pm0.027}$(32)	$0.592_{\pm0.0}$(33)	NAN	$0.546_{\pm0.003}$(35)	$0.746_{\pm0.0}$(26)
fraud	$0.955_{\pm0.001}$(10)	$0.95_{\pm0.006}$(18)	$0.849_{\pm0.001}$(34)	NAN	$0.954_{\pm0.004}$(13)	$0.921_{\pm0.001}$(32)	NAN	$0.487_{\pm0.002}$(42)	$0.951_{\pm0.0}$(15)
http	$0.997_{\pm0.0}$(17)	$0.999_{\pm0.0}$(6)	$0.731_{\pm0.0}$(34)	NAN	$0.997_{\pm0.003}$(17)	$0.193_{\pm0.001}$(39)	NAN	$0.397_{\pm0.002}$(38)	$0.995_{\pm0.0}$(21)
magic.gamma	$0.71_{\pm0.006}$(27)	$0.759_{\pm0.017}$(18)	$0.758_{\pm0.002}$(19)	$0.619_{\pm0.005}$(40)	$0.695_{\pm0.011}$(31)	$0.771_{\pm0.001}$(13)	$0.53_{\pm0.008}$(44)	$0.661_{\pm0.004}$(35)	$0.642_{\pm0.001}$(36)
mammography	$0.721_{\pm0.013}$(38)	$0.864_{\pm0.014}$(14)	$0.514_{\pm0.007}$(44)	$0.69_{\pm0.005}$(40)	$0.822_{\pm0.011}$(28)	$0.823_{\pm0.004}$(26)	$0.43_{\pm0.034}$(46)	$0.664_{\pm0.006}$(43)	$0.885_{\pm0.005}$(4)
shuttle	$0.72_{\pm0.002}$(36)	$0.997_{\pm0.001}$(11)	$0.496_{\pm0.006}$(42)	NAN	$0.942_{\pm0.033}$(32)	$0.499_{\pm0.012}$(41)	NAN	$0.551_{\pm0.001}$(39)	$0.988_{\pm0.0}$(27)
skin	$0.624_{\pm0.004}$(27)	$0.906_{\pm0.01}$(9)	$0.494_{\pm0.001}$(35)	NAN	$0.461_{\pm0.086}$(37)	$0.537_{\pm0.001}$(32)	NAN	$0.539_{\pm0.001}$(31)	$0.336_{\pm0.002}$(41)
smtp	$0.723_{\pm0.032}$(37)	$0.912_{\pm0.022}$(8)	$0.874_{\pm0.003}$(18)	NAN	$0.918_{\pm0.005}$(7)	$0.897_{\pm0.015}$(13)	NAN	$0.87_{\pm0.025}$(19)	$0.807_{\pm0.002}$(31)
Avg Ranking	29.18	13.36	31.55	40.0	22.09	27.73	45.0	33.36	22.91

Table 11: AUPRC results on 11 large datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank).

Dataset	TCCM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
ALOI	$0.085_{\pm0.005}$(11)	$0.077_{\pm0.001}$(15)	$0.075_{\pm0.002}$(16)	$0.061_{\pm0.001}$(40)	$0.068_{\pm0.001}$(29)	$0.061_{\pm0.003}$(40)	$0.072_{\pm0.002}$(20)	$0.079_{\pm0.003}$(14)	$0.059_{\pm0.0}$(44)	$0.066_{\pm0.001}$(31)	$0.069_{\pm0.001}$(27)	$0.099_{\pm0.003}$(8)
celeba	$0.136_{\pm0.039}$(16)	$0.106_{\pm0.013}$(21)	$0.199_{\pm0.002}$(2)	$0.046_{\pm0.002}$(37)	$0.144_{\pm0.026}$(13)	$0.066_{\pm0.028}$(33)	$0.141_{\pm0.065}$(14)	$0.088_{\pm0.008}$(28)	$0.044_{\pm0.0}$(38)	$0.135_{\pm0.017}$(17)	$0.095_{\pm0.006}$(25)	$0.084_{\pm0.012}$(29)
cover	$0.835_{\pm0.028}$(3)	$0.477_{\pm0.135}$(10)	$0.393_{\pm0.012}$(12)	$0.021_{\pm0.002}$(40)	$0.072_{\pm0.039}$(30)	$0.124_{\pm0.081}$(25)	$0.138_{\pm0.15}$(24)	$0.494_{\pm0.075}$(9)	$0.019_{\pm0.0}$(42)	$0.721_{\pm0.028}$(8)	$0.045_{\pm0.005}$(33)	$0.824_{\pm0.024}$(4)
donors	$0.96_{\pm0.056}$(6)	$0.474_{\pm0.055}$(12)	$0.122_{\pm0.008}$(36)	$0.141_{\pm0.012}$(35)	$0.215_{\pm0.165}$(30)	$0.34_{\pm0.214}$(19)	$0.317_{\pm0.101}$(22)	$0.328_{\pm0.026}$(20)	$0.112_{\pm0.0}$(38)	$0.666_{\pm0.054}$(10)	$0.293_{\pm0.018}$(26)	$0.986_{\pm0.01}$(3)
fraud	$0.66_{\pm0.132}$(5)	$0.627_{\pm0.029}$(8)	$0.345_{\pm0.014}$(21)	$0.006_{\pm0.0}$(39)	$0.29_{\pm0.016}$(25)	$0.053_{\pm0.089}$(35)	$0.25_{\pm0.095}$(29)	$0.538_{\pm0.026}$(13)	$0.003_{\pm0.0}$(42)	$0.742_{\pm0.007}$(2)	$0.723_{\pm0.009}$(3)	$0.743_{\pm0.017}$(1)
http	$0.99_{\pm0.015}$(5)	$0.875_{\pm0.073}$(15)	$0.418_{\pm0.467}$(28)	$0.039_{\pm0.035}$(33)	$0.9_{\pm0.017}$(13)	$0.456_{\pm0.212}$(27)	$0.817_{\pm0.137}$(17)	$0.515_{\pm0.003}$(25)	$0.008_{\pm0.0}$(41)	$0.579_{\pm0.022}$(22)	$0.4_{\pm0.255}$(29)	$0.102_{\pm0.056}$(31)
magic.gamma	$0.864_{\pm0.008}$(8)	$0.856_{\pm0.006}$(10)	$0.745_{\pm0.003}$(31)	$0.567_{\pm0.014}$(43)	$0.73_{\pm0.04}$(33)	$0.669_{\pm0.065}$(38)	$0.729_{\pm0.029}$(34)	$0.81_{\pm0.004}$(14)	$0.829_{\pm0.008}$(12)	$0.895_{\pm0.001}$(1)	$0.737_{\pm0.006}$(32)	$0.887_{\pm0.005}$(2)
mammography	$0.439_{\pm0.04}$(6)	$0.418_{\pm0.089}$(10)	$0.35_{\pm0.108}$(19)	$0.054_{\pm0.018}$(44)	$0.477_{\pm0.046}$(2)	$0.426_{\pm0.073}$(7)	$0.345_{\pm0.073}$(23)	$0.361_{\pm0.023}$(17)	$0.251_{\pm0.033}$(31)	$0.37_{\pm0.038}$(16)	$0.134_{\pm0.009}$(41)	$0.36_{\pm0.039}$(18)
shuttle	$0.99_{\pm0.002}$(5)	$0.974_{\pm0.005}$(14)	$0.957_{\pm0.006}$(23)	$0.311_{\pm0.07}$(38)	$0.967_{\pm0.009}$(16)	$0.894_{\pm0.089}$(31)	$0.959_{\pm0.009}$(22)	$0.98_{\pm0.003}$(10)	$0.133_{\pm0.0}$(42)	$0.938_{\pm0.006}$(29)	$0.969_{\pm0.006}$(15)	$0.996_{\pm0.002}$(2)
skin	$0.732_{\pm0.119}$(6)	$0.635_{\pm0.111}$(17)	$0.358_{\pm0.002}$(32)	$0.377_{\pm0.039}$(28)	$0.43_{\pm0.076}$(26)	$0.587_{\pm0.083}$(19)	$0.392_{\pm0.084}$(27)	$0.549_{\pm0.004}$(21)	$0.716_{\pm0.008}$(8)	$0.698_{\pm0.004}$(11)	$0.719_{\pm0.01}$(7)	$0.972_{\pm0.003}$(3)
smtp	$0.507_{\pm0.102}$(8)	$0.462_{\pm0.122}$(13)	$0.467_{\pm0.045}$(12)	$0.007_{\pm0.005}$(37)	$0.33_{\pm0.095}$(22)	$0.214_{\pm0.214}$(23)	$0.41_{\pm0.082}$(20)	$0.616_{\pm0.064}$(4)	$0.001_{\pm0.0}$(40)	$0.471_{\pm0.042}$(11)	$0.143_{\pm0.143}$(26)	$0.102_{\pm0.092}$(28)
Avg Ranking	7.18	13.18	21.09	37.64	21.73	27.0	22.91	15.91	34.36	14.36	24.0	11.73

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
ALOI	$0.087_{\pm0.004}$(9)	$0.133_{\pm0.002}$(5)	$0.086_{\pm0.007}$(10)	$0.064_{\pm0.002}$(35)	$0.082_{\pm0.003}$(12)	$0.219_{\pm0.004}$(1)	$0.075_{\pm0.003}$(16)	$0.069_{\pm0.002}$(27)	$0.06_{\pm0.003}$(43)	$0.072_{\pm0.001}$(20)	$0.068_{\pm0.001}$(29)	$0.072_{\pm0.001}$(20)
celeba	$0.14_{\pm0.059}$(15)	$0.09_{\pm0.001}$(26)	$0.033_{\pm0.004}$(42)	NAN	$0.1_{\pm0.008}$(23)	$0.068_{\pm0.001}$(31)	$0.132_{\pm0.016}$(19)	$0.071_{\pm0.011}$(30)	NAN	$0.067_{\pm0.003}$(32)	$0.048_{\pm0.002}$(36)	$0.197_{\pm0.001}$(3)
cover	$0.891_{\pm0.013}$(2)	$0.814_{\pm0.009}$(6)	$0.196_{\pm0.37}$(18)	NAN	$0.414_{\pm0.209}$(11)	$0.903_{\pm0.009}$(1)	$0.098_{\pm0.065}$(27)	$0.026_{\pm0.001}$(38)	NAN	$0.073_{\pm0.009}$(29)	$0.021_{\pm0.001}$(40)	$0.283_{\pm0.003}$(14)
donors	$0.566_{\pm0.301}$(11)	$0.984_{\pm0.001}$(5)	$0.362_{\pm0.261}$(18)	NAN	$0.995_{\pm0.003}$(2)	$0.999_{\pm0.0}$(1)	$0.956_{\pm0.03}$(7)	$0.059_{\pm0.001}$(41)	NAN	NAN	$0.081_{\pm0.019}$(40)	$0.281_{\pm0.046}$(27)
fraud	$0.636_{\pm0.073}$(7)	$0.373_{\pm0.012}$(17)	$0.521_{\pm0.173}$(14)	NAN	$0.638_{\pm0.035}$(6)	$0.573_{\pm0.029}$(12)	$0.693_{\pm0.044}$(4)	$0.008_{\pm0.0}$(38)	NAN	$0.349_{\pm0.033}$(19)	$0.004_{\pm0.0}$(40)	$0.273_{\pm0.008}$(26)
http	$0.995_{\pm0.005}$(3)	$0.995_{\pm0.001}$(3)	$0.021_{\pm0.014}$(34)	NAN	$0.961_{\pm0.057}$(6)	$0.821_{\pm0.248}$(16)	$0.78_{\pm0.222}$(18)	$0.009_{\pm0.002}$(40)	NAN	NAN	$0.015_{\pm0.003}$(35)	$0.911_{\pm0.004}$(9)
magic.gamma	$0.876_{\pm0.017}$(3)	$0.863_{\pm0.001}$(9)	$0.647_{\pm0.016}$(41)	$0.691_{\pm0.009}$(36)	$0.819_{\pm0.005}$(13)	$0.876_{\pm0.003}$(3)	$0.872_{\pm0.031}$(5)	$0.522_{\pm0.05}$(46)	$0.798_{\pm0.009}$(19)	$0.779_{\pm0.004}$(23)	$0.539_{\pm0.029}$(45)	$0.764_{\pm0.012}$(26)
mammography	$0.347_{\pm0.08}$(21)	$0.419_{\pm0.025}$(9)	$0.339_{\pm0.219}$(24)	$0.279_{\pm0.025}$(26)	$0.182_{\pm0.028}$(36)	$0.459_{\pm0.019}$(3)	$0.346_{\pm0.124}$(22)	$0.26_{\pm0.052}$(28)	$0.169_{\pm0.056}$(38)	$0.203_{\pm0.064}$(34)	$0.182_{\pm0.02}$(36)	$0.375_{\pm0.025}$(14)
shuttle	$0.996_{\pm0.002}$(2)	$0.989_{\pm0.002}$(6)	$0.945_{\pm0.043}$(27)	$0.952_{\pm0.009}$(24)	$0.996_{\pm0.003}$(2)	$0.997_{\pm0.002}$(1)	$0.966_{\pm0.004}$(18)	$0.074_{\pm0.005}$(44)	$0.499_{\pm0.128}$(35)	$0.98_{\pm0.004}$(10)	$0.077_{\pm0.015}$(43)	$0.962_{\pm0.005}$(21)
skin	$0.95_{\pm0.023}$(4)	$0.996_{\pm0.001}$(1)	$0.295_{\pm0.016}$(40)	NAN	$0.314_{\pm0.018}$(36)	$0.975_{\pm0.001}$(2)	$0.469_{\pm0.12}$(24)	$0.345_{\pm0.072}$(33)	NAN	$0.854_{\pm0.01}$(5)	$0.293_{\pm0.039}$(41)	$0.373_{\pm0.001}$(29)
smtp	$0.07_{\pm0.084}$(30)	$0.553_{\pm0.05}$(6)	$0.001_{\pm0.0}$(40)	$0.454_{\pm0.053}$(15)	$0.042_{\pm0.04}$(31)	$0.662_{\pm0.009}$(1)	$0.454_{\pm0.055}$(15)	$0.001_{\pm0.0}$(40)	NAN	$0.473_{\pm0.049}$(10)	$0.001_{\pm0.0}$(40)	$0.46_{\pm0.054}$(14)
Avg Ranking	9.73	8.45	28.0	27.2	16.18	6.55	15.91	36.82	33.75	20.22	38.64	18.45

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
ALOI	$0.072_{\pm0.001}$(20)	$0.065_{\pm0.0}$(32)	$0.064_{\pm0.0}$(35)	$0.17_{\pm0.006}$(2)	$0.074_{\pm0.001}$(19)	$0.065_{\pm0.0}$(32)	$0.065_{\pm0.0}$(32)	$0.102_{\pm0.002}$(7)	$0.064_{\pm0.001}$(35)	$0.061_{\pm0.004}$(40)	$0.063_{\pm0.002}$(38)	$0.075_{\pm0.001}$(16)
celeba	$0.17_{\pm0.05}$(7)	$0.097_{\pm0.002}$(24)	$0.171_{\pm0.001}$(6)	$0.049_{\pm0.003}$(35)	$0.166_{\pm0.001}$(9)	$0.17_{\pm0.001}$(7)	$0.134_{\pm0.008}$(18)	$0.089_{\pm0.001}$(27)	$0.15_{\pm NAN}$(11)	$0.113_{\pm0.066}$(20)	$0.149_{\pm0.046}$(12)	$0.203_{\pm0.001}$(1)
cover	$0.168_{\pm0.022}$(21)	$0.038_{\pm0.005}$(34)	$0.187_{\pm0.002}$(20)	$0.818_{\pm0.09}$(5)	$0.197_{\pm0.003}$(17)	$0.051_{\pm0.001}$(32)	$0.081_{\pm0.012}$(28)	$0.359_{\pm0.006}$(13)	$0.155_{\pm0.082}$(22)	$0.208_{\pm0.114}$(16)	$0.031_{\pm0.0}$(36)	$0.223_{\pm0.004}$(15)
donors	$0.447_{\pm0.02}$(13)	$0.114_{\pm0.016}$(37)	$0.414_{\pm0.001}$(16)	$0.919_{\pm0.041}$(8)	$0.432_{\pm0.001}$(14)	$0.3_{\pm0.038}$(25)	$0.397_{\pm0.027}$(17)	$0.713_{\pm0.003}$(9)	$0.28_{\pm NAN}$(28)	$0.309_{\pm0.28}$(24)	$0.31_{\pm0.122}$(23)	$0.428_{\pm0.001}$(15)
fraud	$0.257_{\pm0.008}$(28)	$0.242_{\pm0.042}$(30)	$0.331_{\pm0.006}$(22)	$0.016_{\pm0.004}$(37)	$0.587_{\pm0.015}$(11)	$0.349_{\pm0.018}$(19)	$0.208_{\pm0.055}$(33)	$0.389_{\pm0.012}$(16)	$0.321_{\pm NAN}$(23)	$0.311_{\pm0.126}$(24)	$0.601_{\pm0.022}$(10)	$0.352_{\pm0.01}$(18)
http	$0.902_{\pm0.002}$(12)	$0.551_{\pm0.003}$(24)	$0.252_{\pm0.001}$(30)	$0.069_{\pm0.004}$(32)	$0.911_{\pm0.003}$(9)	$0.562_{\pm0.159}$(23)	$0.493_{\pm0.12}$(26)	$0.997_{\pm0.003}$(2)	$0.908_{\pm0.006}$(11)	$0.01_{\pm0.003}$(39)	$0.918_{\pm0.005}$(7)	$0.998_{\pm0.002}$(1)
magic.gamma	$0.798_{\pm0.007}$(19)	$0.755_{\pm0.002}$(28)	$0.68_{\pm0.002}$(37)	$0.868_{\pm0.002}$(7)	$0.834_{\pm0.001}$(11)	$0.773_{\pm0.002}$(24)	$0.802_{\pm0.008}$(17)	$0.809_{\pm0.001}$(15)	$0.668_{\pm0.016}$(39)	$0.758_{\pm0.012}$(27)	$0.772_{\pm0.003}$(25)	$0.791_{\pm0.001}$(21)
mammography	$0.348_{\pm0.015}$(20)	$0.193_{\pm0.01}$(35)	$0.549_{\pm0.008}$(1)	$0.257_{\pm0.025}$(29)	$0.404_{\pm0.021}$(11)	$0.209_{\pm0.005}$(33)	$0.373_{\pm0.071}$(15)	$0.425_{\pm0.018}$(8)	$0.458_{\pm0.013}$(4)	$0.445_{\pm0.044}$(5)	$0.08_{\pm0.012}$(43)	$0.403_{\pm0.016}$(12)
shuttle	$0.965_{\pm0.005}$(19)	$0.456_{\pm0.028}$(37)	$0.95_{\pm0.003}$(26)	$0.478_{\pm0.274}$(36)	$0.967_{\pm0.005}$(16)	$0.976_{\pm0.003}$(12)	$0.985_{\pm0.007}$(7)	$0.981_{\pm0.003}$(8)	$0.952_{\pm0.014}$(24)	$0.5_{\pm0.336}$(34)	$0.905_{\pm0.006}$(30)	$0.975_{\pm0.002}$(13)
skin	$0.712_{\pm0.034}$(9)	$0.448_{\pm0.002}$(25)	$0.303_{\pm0.0}$(37)	$0.578_{\pm0.018}$(20)	$0.645_{\pm0.001}$(15)	$0.534_{\pm0.002}$(22)	$0.638_{\pm0.008}$(16)	$0.65_{\pm0.002}$(14)	$0.297_{\pm0.032}$(39)	$0.515_{\pm0.075}$(23)	$0.625_{\pm0.002}$(18)	$0.664_{\pm0.001}$(13)
smtp	$0.453_{\pm0.061}$(18)	$0.16_{\pm0.055}$(25)	$0.585_{\pm0.037}$(5)	$0.003_{\pm0.0}$(38)	$0.367_{\pm0.041}$(21)	$0.009_{\pm0.001}$(35)	$0.009_{\pm0.001}$(35)	$0.633_{\pm0.051}$(3)	$0.183_{\pm0.165}$(24)	$0.086_{\pm0.053}$(29)	$0.01_{\pm0.0}$(34)	$0.636_{\pm0.047}$(2)
Avg Ranking	16.91	30.09	21.36	22.64	13.91	24.0	22.18	11.09	23.64	25.55	25.09	11.55

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
ALOI	$0.062_{\pm0.0}$(39)	$0.071_{\pm0.001}$(25)	$0.158_{\pm0.001}$(3)	NAN	$0.072_{\pm0.001}$(20)	$0.115_{\pm0.002}$(6)	NAN	$0.142_{\pm0.002}$(4)	$0.07_{\pm0.001}$(26)
celeba	$0.044_{\pm0.0}$(38)	$0.181_{\pm0.031}$(4)	$0.041_{\pm0.0}$(40)	NAN	$0.106_{\pm0.004}$(21)	$0.064_{\pm0.0}$(34)	NAN	$0.038_{\pm0.001}$(41)	$0.181_{\pm0.001}$(4)
cover	$0.071_{\pm0.001}$(31)	$0.144_{\pm0.03}$(23)	$0.028_{\pm0.0}$(37)	NAN	$0.189_{\pm0.007}$(19)	$0.033_{\pm0.0}$(35)	NAN	$0.022_{\pm0.0}$(39)	$0.117_{\pm0.001}$(26)
donors	$0.169_{\pm0.002}$(31)	$0.319_{\pm0.063}$(21)	$0.101_{\pm0.0}$(39)	NAN	$0.143_{\pm0.007}$(34)	$0.157_{\pm0.0}$(32)	NAN	$0.145_{\pm0.002}$(33)	$0.224_{\pm0.0}$(29)
fraud	$0.416_{\pm0.021}$(15)	$0.268_{\pm0.009}$(27)	$0.02_{\pm0.0}$(36)	NAN	$0.219_{\pm0.111}$(32)	$0.08_{\pm0.001}$(34)	NAN	$0.004_{\pm0.0}$(40)	$0.228_{\pm0.003}$(31)
http	$0.708_{\pm0.014}$(20)	$0.893_{\pm0.002}$(14)	$0.014_{\pm0.0}$(36)	NAN	$0.747_{\pm0.207}$(19)	$0.014_{\pm0.0}$(36)	NAN	$0.012_{\pm0.0}$(38)	$0.602_{\pm0.003}$(21)
magic.gamma	$0.749_{\pm0.006}$(30)	$0.8_{\pm0.009}$(18)	$0.788_{\pm0.002}$(22)	$0.635_{\pm0.005}$(42)	$0.753_{\pm0.008}$(29)	$0.808_{\pm0.001}$(16)	$0.56_{\pm0.006}$(44)	$0.657_{\pm0.005}$(40)	$0.703_{\pm0.001}$(35)
mammography	$0.14_{\pm0.005}$(40)	$0.378_{\pm0.056}$(13)	$0.043_{\pm0.001}$(45)	$0.11_{\pm0.003}$(42)	$0.254_{\pm0.02}$(30)	$0.23_{\pm0.013}$(32)	$0.038_{\pm0.002}$(46)	$0.143_{\pm0.007}$(39)	$0.271_{\pm0.006}$(27)
shuttle	$0.684_{\pm0.014}$(32)	$0.965_{\pm0.007}$(19)	$0.171_{\pm0.002}$(41)	NAN	$0.633_{\pm0.12}$(33)	$0.203_{\pm0.002}$(40)	NAN	$0.211_{\pm0.002}$(39)	$0.942_{\pm0.004}$(28)
skin	$0.366_{\pm0.002}$(30)	$0.702_{\pm0.017}$(10)	$0.342_{\pm0.0}$(34)	NAN	$0.298_{\pm0.031}$(38)	$0.332_{\pm0.0}$(35)	NAN	$0.359_{\pm0.001}$(31)	$0.253_{\pm0.001}$(42)
smtp	$0.022_{\pm0.008}$(32)	$0.451_{\pm0.052}$(19)	$0.002_{\pm0.0}$(39)	NAN	$0.488_{\pm0.113}$(9)	$0.105_{\pm0.014}$(27)	NAN	$0.017_{\pm0.01}$(33)	$0.454_{\pm0.054}$(15)
Avg Ranking	30.73	17.55	33.82	42.0	25.82	29.73	45.0	34.27	25.82

Table 12: AUROC results on 9 high-dimensional datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank).

Dataset	TCCM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
backdoor	$0.948_{\pm0.014}(5)$	$0.935_{\pm0.002}(7)$	$0.919_{\pm0.002}(11)$	$0.488_{\pm0.139}(38)$	$0.58_{\pm0.15}(31)$	$0.451_{\pm0.15}(39)$	$0.551_{\pm0.043}(33)$	$0.926_{\pm0.001}(9)$	$0.5_{\pm0.0}(34)$	$0.918_{\pm0.004}(12)$	$0.606_{\pm0.008}(29)$	$0.925_{\pm0.007}(10)$
campaign	$0.785_{\pm0.017}(8)$	$0.815_{\pm0.011}(1)$	$0.79_{\pm0.009}(6)$	$0.527_{\pm0.041}(42)$	$0.758_{\pm0.021}(20)$	$0.602_{\pm0.04}(39)$	$0.731_{\pm0.025}(25)$	$0.676_{\pm0.01}(35)$	$0.59_{\pm0.123}(40)$	$0.787_{\pm0.005}(7)$	$0.708_{\pm0.007}(28)$	$0.701_{\pm0.063}(30)$
census	$0.715_{\pm0.004}(4)$	$0.721_{\pm0.003}(3)$	$0.705_{\pm0.005}(10)$	$0.407_{\pm0.062}(37)$	$0.685_{\pm0.037}(16)$	$0.512_{\pm0.028}(31)$	$0.693_{\pm0.015}(13)$	$0.579_{\pm0.016}(29)$	$0.5_{\pm0.0}(32)$	$0.693_{\pm0.008}(13)$	$0.637_{\pm0.002}(22)$	$0.493_{\pm0.039}(34)$
InternetAds	$0.872_{\pm0.017}(6)$	$0.883_{\pm0.006}(1)$	$0.882_{\pm0.006}(2)$	$0.373_{\pm0.026}(40)$	$0.595_{\pm0.054}(24)$	NAN	$0.683_{\pm0.019}(21)$	$0.55_{\pm0.018}(28)$	$0.492_{\pm0.07}(35)$	$0.85_{\pm0.03}(10)$	$0.695_{\pm0.044}(20)$	$0.868_{\pm0.018}(7)$
mnist	$0.933_{\pm0.018}(7)$	$0.936_{\pm0.003}(4)$	$0.93_{\pm0.003}(8)$	$0.542_{\pm0.034}(44)$	$0.871_{\pm0.017}(20)$	$0.762_{\pm0.061}(32)$	$0.811_{\pm0.027}(27)$	$0.882_{\pm0.01}(19)$	$0.852_{\pm0.022}(24)$	$0.902_{\pm0.017}(17)$	$0.815_{\pm0.009}(26)$	$0.859_{\pm0.009}(23)$
musk	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.512_{\pm0.1}(41)$	$0.962_{\pm0.035}(30)$	$0.899_{\pm0.061}(34)$	$0.987_{\pm0.024}(29)$	$0.999_{\pm0.001}(25)$	$0.281_{\pm0.177}(44)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$
optdigits	$0.9_{\pm0.038}(7)$	$0.87_{\pm0.004}(11)$	$0.682_{\pm0.033}(23)$	$0.479_{\pm0.043}(38)$	$0.547_{\pm0.112}(30)$	$0.613_{\pm0.202}(28)$	$0.542_{\pm0.176}(32)$	$0.626_{\pm0.066}(25)$	$0.859_{\pm0.049}(13)$	$0.849_{\pm0.023}(15)$	$0.618_{\pm0.02}(27)$	$0.883_{\pm0.011}(10)$
SpamBase	$0.866_{\pm0.006}(1)$	$0.816_{\pm0.011}(14)$	$0.813_{\pm0.012}(15)$	$0.54_{\pm0.01}(37)$	$0.825_{\pm0.01}(10)$	$0.633_{\pm0.034}(35)$	$0.796_{\pm0.031}(22)$	$0.5_{\pm0.029}(38)$	$0.777_{\pm0.006}(25)$	$0.848_{\pm0.011}(5)$	$0.784_{\pm0.011}(23)$	$0.784_{\pm0.016}(23)$
speech	$0.549_{\pm0.009}(5)$	$0.476_{\pm0.007}(27)$	$0.471_{\pm0.007}(35)$	$0.472_{\pm0.035}(33)$	$0.486_{\pm0.028}(23)$	$0.498_{\pm0.034}(17)$	$0.514_{\pm0.035}(14)$	$0.476_{\pm0.028}(27)$	$0.527_{\pm0.042}(9)$	$0.523_{\pm0.009}(12)$	$0.521_{\pm0.061}(13)$	$0.531_{\pm0.029}(7)$
Avg Ranking	4.89	7.67	12.33	38.89	22.67	31.88	24.0	26.11	28.44	10.22	21.0	16.11

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
backdoor	$0.938_{\pm0.01}(6)$	$0.951_{\pm0.001}(3)$	$0.795_{\pm0.057}(18)$	$0.495_{\pm0.077}(37)$	NAN	$0.954_{\pm0.001}(2)$	NAN	$0.859_{\pm0.037}(15)$	NAN	$0.5_{\pm0.0}(34)$	$0.77_{\pm0.123}(19)$	$0.911_{\pm0.001}(13)$
campaign	$0.717_{\pm0.061}(27)$	$0.785_{\pm0.002}(8)$	$0.682_{\pm0.038}(34)$	$0.393_{\pm0.02}(44)$	$0.811_{\pm0.005}(2)$	$0.731_{\pm0.003}(25)$	$0.785_{\pm0.016}(8)$	$0.65_{\pm0.053}(37)$	$0.687_{\pm0.078}(33)$	$0.763_{\pm0.005}(19)$	$0.696_{\pm0.023}(32)$	$0.781_{\pm0.001}(11)$
census	$0.66_{\pm0.014}(20)$	$0.722_{\pm0.001}(2)$	$0.695_{\pm0.022}(12)$	NAN	NAN	$0.674_{\pm0.003}(19)$	NAN	$0.59_{\pm0.051}(26)$	NAN	$0.616_{\pm0.106}(25)$	$0.573_{\pm0.037}(30)$	$0.706_{\pm0.001}(8)$
InternetAds	$0.809_{\pm0.03}(12)$	$0.795_{\pm0.032}(14)$	$0.772_{\pm0.035}(15)$	$0.651_{\pm0.103}(23)$	NAN	$0.856_{\pm0.005}(8)$	NAN	$0.453_{\pm0.04}(36)$	$0.796_{\pm0.023}(13)$	$0.853_{\pm0.03}(9)$	$0.423_{\pm0.058}(37)$	$0.881_{\pm0.006}(3)$
mnist	$0.793_{\pm0.05}(29)$	$0.944_{\pm0.002}(2)$	$0.791_{\pm0.073}(30)$	$0.44_{\pm0.13}(46)$	$0.913_{\pm0.014}(12)$	$0.934_{\pm0.004}(6)$	$0.928_{\pm0.014}(10)$	$0.686_{\pm0.106}(36)$	$0.809_{\pm0.023}(28)$	$0.912_{\pm0.006}(14)$	$0.681_{\pm0.078}(37)$	$0.935_{\pm0.002}(5)$
musk	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.942_{\pm0.055}(32)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.893_{\pm0.132}(35)$	$0.875_{\pm0.103}(37)$	$1.0_{\pm0.0}(1)$	$0.889_{\pm0.162}(36)$	$1.0_{\pm0.0}(1)$
optdigits	$0.868_{\pm0.08}(12)$	$0.961_{\pm0.004}(5)$	$0.724_{\pm0.167}(21)$	$0.802_{\pm0.144}(20)$	$0.981_{\pm0.004}(2)$	$0.997_{\pm0.001}(1)$	$0.852_{\pm0.084}(14)$	$0.245_{\pm0.071}(45)$	$0.429_{\pm0.148}(41)$	$0.942_{\pm0.011}(6)$	$0.26_{\pm0.105}(44)$	$0.686_{\pm0.057}(22)$
SpamBase	$0.744_{\pm0.049}(28)$	$0.851_{\pm0.005}(3)$	$0.825_{\pm0.015}(10)$	$0.405_{\pm0.188}(43)$	$0.843_{\pm0.007}(6)$	$0.83_{\pm0.005}(9)$	$0.82_{\pm0.014}(12)$	$0.431_{\pm0.105}(42)$	$0.837_{\pm0.025}(8)$	$0.851_{\pm0.007}(3)$	$0.398_{\pm0.098}(44)$	$0.812_{\pm0.013}(16)$
speech	$0.445_{\pm0.029}(44)$	$0.567_{\pm0.011}(2)$	$0.492_{\pm0.031}(20)$	$0.504_{\pm0.079}(16)$	$0.565_{\pm0.036}(3)$	$0.57_{\pm0.063}(1)$	$0.472_{\pm0.007}(33)$	$0.458_{\pm0.034}(42)$	$0.493_{\pm0.047}(19)$	$0.511_{\pm0.057}(15)$	$0.457_{\pm0.031}(43)$	$0.471_{\pm0.006}(35)$
Avg Ranking	19.89	4.44	17.89	32.62	4.33	8.0	13.0	34.89	25.57	14.0	35.78	12.67

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
backdoor	$0.714_{\pm0.04}(22)$	$0.744_{\pm0.127}(21)$	$0.846_{\pm0.001}(17)$	$0.95_{\pm0.006}(4)$	$0.928_{\pm0.0}(8)$	$0.713_{\pm0.005}(23)$	$0.746_{\pm0.024}(20)$	$0.905_{\pm0.001}(14)$	NAN	$0.411_{\pm0.226}(40)$	$0.851_{\pm0.102}(16)$	$0.626_{\pm0.004}(28)$
campaign	$0.772_{\pm0.002}(16)$	$0.768_{\pm0.02}(18)$	$0.77_{\pm0.001}(17)$	$0.675_{\pm0.043}(36)$	$0.799_{\pm0.001}(4)$	$0.775_{\pm0.003}(14)$	$0.738_{\pm0.018}(22)$	$0.775_{\pm0.001}(14)$	$0.699_{\pm0.03}(31)$	$0.584_{\pm0.064}(41)$	$0.791_{\pm0.006}(5)$	$0.776_{\pm0.001}(13)$
census	$0.708_{\pm0.003}(6)$	$0.658_{\pm0.174}(21)$	NAN	$0.582_{\pm0.006}(28)$	$0.706_{\pm0.001}(8)$	$0.624_{\pm0.001}(23)$	$0.624_{\pm0.012}(23)$	$0.723_{\pm0.002}(1)$	NAN	$0.483_{\pm0.082}(35)$	NAN	$0.702_{\pm0.001}(11)$
InternetAds	$0.697_{\pm0.044}(19)$	$0.5_{\pm0.0}(32)$	$0.678_{\pm0.007}(22)$	$0.746_{\pm0.041}(16)$	$0.881_{\pm0.016}(3)$	$0.403_{\pm0.05}(39)$	$0.419_{\pm0.079}(38)$	$0.841_{\pm0.019}(11)$	$0.561_{\pm0.058}(26)$	$0.575_{\pm0.122}(25)$	NAN	$0.705_{\pm0.044}(18)$
mnist	$0.913_{\pm0.003}(12)$	$0.605_{\pm0.117}(41)$	$0.747_{\pm0.004}(33)$	$0.929_{\pm0.004}(9)$	$0.926_{\pm0.001}(11)$	$0.611_{\pm0.006}(39)$	$0.864_{\pm0.012}(21)$	$0.945_{\pm0.002}(1)$	$0.722_{\pm0.092}(35)$	$0.728_{\pm0.078}(34)$	$0.891_{\pm0.017}(18)$	$0.91_{\pm0.001}(15)$
musk	$1.0_{\pm0.0}(1)$	$0.671_{\pm0.008}(38)$	$0.954_{\pm0.002}(31)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.931_{\pm0.016}(33)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.999_{\pm0.001}(25)$	$1.0_{\pm0.0}(1)$
optdigits	$0.835_{\pm0.02}(16)$	$0.383_{\pm0.024}(42)$	$0.609_{\pm0.004}(29)$	$0.975_{\pm0.003}(3)$	$0.813_{\pm0.005}(18)$	$0.899_{\pm0.004}(8)$	$0.809_{\pm0.064}(19)$	$0.973_{\pm0.003}(4)$	$0.475_{\pm0.019}(39)$	$0.498_{\pm0.184}(36)$	$0.652_{\pm0.034}(24)$	$0.626_{\pm0.01}(25)$
SpamBase	$0.811_{\pm0.01}(17)$	$0.491_{\pm0.023}(39)$	$0.66_{\pm0.008}(33)$	$0.757_{\pm0.024}(27)$	$0.797_{\pm0.012}(21)$	$0.772_{\pm0.007}(26)$	$0.819_{\pm0.017}(13)$	$0.857_{\pm0.004}(2)$	$0.661_{\pm0.05}(32)$	$0.681_{\pm0.103}(31)$	$0.799_{\pm0.044}(20)$	$0.811_{\pm0.013}(17)$
speech	$0.473_{\pm0.006}(32)$	$0.482_{\pm0.008}(25)$	$0.471_{\pm0.006}(35)$	$0.496_{\pm0.006}(18)$	$0.527_{\pm0.005}(9)$	$0.476_{\pm0.008}(27)$	$0.468_{\pm0.023}(40)$	$0.531_{\pm0.059}(7)$	$0.489_{\pm0.016}(21)$	$0.474_{\pm0.008}(30)$	$0.526_{\pm0.005}(11)$	$0.468_{\pm0.006}(40)$
Avg Ranking	15.67	30.78	27.12	15.78	9.22	22.22	25.44	6.11	26.43	30.33	17.0	18.67

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
backdoor	$0.567_{\pm0.002}(32)$	$0.678_{\pm0.021}(24)$	$0.668_{\pm0.001}(25)$	NAN	$0.606_{\pm0.025}(29)$	$0.666_{\pm0.001}(26)$	NAN	$0.664_{\pm0.001}(27)$	$0.5_{\pm0.0}(34)$
campaign	$0.805_{\pm0.006}(3)$	$0.74_{\pm0.028}(21)$	$0.706_{\pm0.002}(29)$	NAN	$0.738_{\pm0.003}(22)$	$0.736_{\pm0.001}(24)$	NAN	$0.631_{\pm0.003}(38)$	$0.5_{\pm0.0}(43)$
census	$0.681_{\pm0.001}(18)$	$0.707_{\pm0.006}(7)$	$0.434_{\pm0.001}(36)$	NAN	$0.689_{\pm0.005}(15)$	$0.682_{\pm0.001}(17)$	NAN	$0.585_{\pm0.001}(27)$	$0.5_{\pm0.0}(32)$
InternetAds	$0.5_{\pm0.001}(32)$	$0.707_{\pm0.042}(17)$	$0.528_{\pm0.022}(30)$	$0.266_{\pm0.034}(41)$	$0.548_{\pm0.04}(29)$	$0.561_{\pm0.041}(26)$	$0.505_{\pm0.039}(31)$	$0.223_{\pm0.025}(42)$	$0.493_{\pm0.007}(34)$
mnist	$0.639_{\pm0.007}(38)$	$0.904_{\pm0.014}(16)$	$0.766_{\pm0.003}(31)$	$0.594_{\pm0.009}(42)$	$0.863_{\pm0.008}(22)$	$0.844_{\pm0.002}(25)$	$0.546_{\pm0.018}(43)$	$0.61_{\pm0.003}(40)$	$0.5_{\pm0.0}(45)$
musk	$0.036_{\pm0.007}(46)$	$1.0_{\pm0.0}(1)$	$0.056_{\pm0.002}(45)$	$0.449_{\pm0.046}(43)$	$0.997_{\pm0.001}(28)$	$0.499_{\pm0.003}(42)$	$0.557_{\pm0.17}(40)$	$0.623_{\pm0.007}(39)$	$0.999_{\pm0.0}(25)$
optdigits	$0.133_{\pm0.003}(46)$	$0.814_{\pm0.011}(17)$	$0.45_{\pm0.005}(40)$	$0.512_{\pm0.008}(33)$	$0.506_{\pm0.026}(34)$	$0.292_{\pm0.003}(43)$	$0.498_{\pm0.095}(36)$	$0.544_{\pm0.008}(31)$	$0.5_{\pm0.0}(35)$
SpamBase	$0.696_{\pm0.037}(29)$	$0.806_{\pm0.01}(19)$	$0.47_{\pm0.007}(40)$	$0.365_{\pm0.013}(45)$	$0.688_{\pm0.017}(30)$	$0.64_{\pm0.01}(34)$	$0.59_{\pm0.02}(36)$	$0.352_{\pm0.012}(46)$	$0.469_{\pm0.006}(41)$
speech	$0.413_{\pm0.007}(46)$	$0.483_{\pm0.014}(24)$	$0.557_{\pm0.015}(4)$	$0.416_{\pm0.021}(45)$	$0.474_{\pm0.008}(30)$	$0.469_{\pm0.008}(39)$	$0.489_{\pm0.039}(21)$	$0.482_{\pm0.009}(25)$	$0.47_{\pm0.007}(38)$
Avg Ranking	32.22	16.22	31.11	41.5	26.56	30.67	34.5	35.0	36.33

Table 13: AUPRC results on 9 high-dimensional datasets, where we compare TCCM to 44 baselines (with 5 independent runs). We report the mean ± std (rank).

Dataset	TCCM (Ours)	AE	AE-1SVM	ALAD	AnoGAN	DAGMM	DeepSVDD	DIF	DROCC	DTE-Cat	DTE-DDPM	DTE-Gaussian
backdoor	$0.835_{\pm0.063}(8)$	$0.868_{\pm0.002}(4)$	$0.86_{\pm0.006}(5)$	$0.059_{\pm0.03}(35)$	$0.071_{\pm0.033}(32)$	$0.051_{\pm0.014}(36)$	$0.069_{\pm0.012}(33)$	$0.696_{\pm0.03}(10)$	$0.048_{\pm0.0}(38)$	$0.631_{\pm0.037}(12)$	$0.086_{\pm0.002}(26)$	$0.804_{\pm0.017}(9)$
campaign	$0.49_{\pm0.019}(15)$	$0.504_{\pm0.021}(3)$	$0.501_{\pm0.013}(5)$	$0.229_{\pm0.039}(42)$	$0.492_{\pm0.022}(13)$	$0.325_{\pm0.054}(37)$	$0.449_{\pm0.035}(23)$	$0.381_{\pm0.018}(34)$	$0.297_{\pm0.13}(41)$	$0.495_{\pm0.014}(11)$	$0.443_{\pm0.009}(25)$	$0.41_{\pm0.06}(29)$
census	$0.213_{\pm0.005}(5)$	$0.217_{\pm0.004}(2)$	$0.206_{\pm0.006}(8)$	$0.096_{\pm0.018}(36)$	$0.178_{\pm0.016}(19)$	$0.13_{\pm0.014}(31)$	$0.194_{\pm0.014}(14)$	$0.126_{\pm0.005}(32)$	$0.117_{\pm0.0}(34)$	$0.177_{\pm0.008}(20)$	$0.18_{\pm0.002}(18)$	$0.143_{\pm0.013}(26)$
InternetAds	$0.829_{\pm0.042}(4)$	$0.869_{\pm0.011}(1)$	$0.868_{\pm0.011}(2)$	$0.255_{\pm0.006}(40)$	$0.433_{\pm0.114}(24)$	NAN	$0.486_{\pm0.05}(23)$	$0.345_{\pm0.01}(31)$	$0.405_{\pm0.07}(25)$	$0.725_{\pm0.091}(11)$	$0.531_{\pm0.109}(20)$	$0.803_{\pm0.056}(6)$
mnist	$0.747_{\pm0.048}(5)$	$0.737_{\pm0.009}(6)$	$0.716_{\pm0.016}(11)$	$0.212_{\pm0.033}(45)$	$0.622_{\pm0.019}(21)$	$0.494_{\pm0.088}(31)$	$0.55_{\pm0.038}(25)$	$0.624_{\pm0.021}(20)$	$0.636_{\pm0.03}(19)$	$0.644_{\pm0.029}(18)$	$0.557_{\pm0.012}(23)$	$0.666_{\pm0.021}(17)$
musk	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.092_{\pm0.082}(44)$	$0.745_{\pm0.181}(30)$	$0.641_{\pm0.194}(32)$	$0.881_{\pm0.178}(29)$	$0.989_{\pm0.017}(25)$	$0.135_{\pm0.13}(42)$	$1.0_{\pm0.0}(1)$	$0.997_{\pm0.003}(22)$	$1.0_{\pm0.0}(1)$
optdigits	$0.267_{\pm0.092}(9)$	$0.175_{\pm0.004}(15)$	$0.08_{\pm0.009}(25)$	$0.053_{\pm0.006}(38)$	$0.058_{\pm0.012}(34)$	$0.092_{\pm0.068}(23)$	$0.066_{\pm0.03}(31)$	$0.074_{\pm0.012}(27)$	$0.201_{\pm0.055}(12)$	$0.174_{\pm0.024}(16)$	$0.075_{\pm0.002}(26)$	$0.241_{\pm0.018}(11)$
SpamBase	$0.899_{\pm0.004}(1)$	$0.849_{\pm0.01}(14)$	$0.849_{\pm0.011}(14)$	$0.619_{\pm0.009}(37)$	$0.863_{\pm0.007}(9)$	$0.699_{\pm0.022}(33)$	$0.836_{\pm0.028}(21)$	$0.569_{\pm0.026}(41)$	$0.829_{\pm0.009}(22)$	$0.87_{\pm0.011}(7)$	$0.829_{\pm0.01}(22)$	$0.822_{\pm0.013}(24)$
speech	$0.058_{\pm0.005}(2)$	$0.038_{\pm0.002}(22)$	$0.038_{\pm0.001}(22)$	$0.033_{\pm0.002}(38)$	$0.041_{\pm0.006}(16)$	$0.04_{\pm0.006}(17)$	$0.034_{\pm0.002}(36)$	$0.034_{\pm0.003}(36)$	$0.04_{\pm0.007}(17)$	$0.047_{\pm0.005}(9)$	$0.04_{\pm0.004}(17)$	$0.049_{\pm0.007}(7)$
Avg Ranking	5.56	7.56	10.33	39.44	22.0	30.0	26.11	28.44	27.78	11.67	22.11	14.44

Dataset	DTE-IG	DTE-NP	GANomaly	GOAD	ICL	LUNAR	MCM	MO_GAAL	PlanarFlow	SLAD	SO_GAAL	VAE
backdoor	$0.851_{\pm0.006}(7)$	$0.642_{\pm0.003}(11)$	$0.153_{\pm0.093}(19)$	$0.062_{\pm0.01}(34)$	NAN	$0.899_{\pm0.004}(1)$	NAN	$0.259_{\pm0.075}(15)$	NAN	$0.048_{\pm0.0}(38)$	$0.13_{\pm0.072}(20)$	$0.856_{\pm0.001}(6)$
campaign	$0.447_{\pm0.025}(24)$	$0.492_{\pm0.003}(13)$	$0.385_{\pm0.058}(33)$	$0.176_{\pm0.01}(44)$	$0.498_{\pm0.009}(7)$	$0.435_{\pm0.002}(26)$	$0.493_{\pm0.01}(12)$	$0.369_{\pm0.052}(35)$	$0.397_{\pm0.085}(31)$	$0.485_{\pm0.007}(17)$	$0.398_{\pm0.027}(30)$	$0.499_{\pm0.003}(6)$
census	$0.188_{\pm0.003}(15)$	$0.217_{\pm0.001}(2)$	$0.188_{\pm0.012}(15)$	NAN	NAN	$0.165_{\pm0.001}(23)$	NAN	$0.154_{\pm0.017}(24)$	NAN	$0.168_{\pm0.046}(22)$	$0.146_{\pm0.013}(25)$	$0.2_{\pm0.003}(11)$
InternetAds	$0.722_{\pm0.06}(12)$	$0.661_{\pm0.1}(13)$	$0.639_{\pm0.103}(14)$	$0.505_{\pm0.078}(22)$	NAN	$0.771_{\pm0.025}(9)$	NAN	$0.365_{\pm0.066}(29)$	$0.61_{\pm0.076}(16)$	$0.735_{\pm0.085}(10)$	$0.356_{\pm0.07}(30)$	$0.867_{\pm0.012}(3)$
mnist	$0.549_{\pm0.091}(26)$	$0.769_{\pm0.02}(3)$	$0.495_{\pm0.072}(30)$	$0.264_{\pm0.06}(43)$	$0.715_{\pm0.039}(13)$	$0.775_{\pm0.023}(1)$	$0.717_{\pm0.031}(10)$	$0.433_{\pm0.095}(34)$	$0.529_{\pm0.03}(29)$	$0.728_{\pm0.021}(7)$	$0.414_{\pm0.059}(36)$	$0.726_{\pm0.006}(8)$
musk	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.714_{\pm0.283}(31)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.6_{\pm0.304}(34)$	$0.546_{\pm0.263}(36)$	$1.0_{\pm0.0}(1)$	$0.598_{\pm0.323}(35)$	$1.0_{\pm0.0}(1)$
optdigits	$0.282_{\pm0.117}(8)$	$0.406_{\pm0.026}(5)$	$0.113_{\pm0.046}(21)$	$0.193_{\pm0.132}(13)$	$0.606_{\pm0.04}(2)$	$0.926_{\pm0.036}(1)$	$0.181_{\pm0.079}(14)$	$0.035_{\pm0.003}(45)$	$0.048_{\pm0.012}(41)$	$0.34_{\pm0.041}(7)$	$0.036_{\pm0.005}(43)$	$0.081_{\pm0.017}(24)$
SpamBase	$0.796_{\pm0.041}(26)$	$0.875_{\pm0.005}(5)$	$0.859_{\pm0.011}(10)$	$0.555_{\pm0.145}(42)$	$0.884_{\pm0.004}(3)$	$0.856_{\pm0.007}(11)$	$0.854_{\pm0.012}(12)$	$0.601_{\pm0.068}(39)$	$0.875_{\pm0.022}(5)$	$0.888_{\pm0.008}(2)$	$0.577_{\pm0.066}(40)$	$0.845_{\pm0.012}(18)$
speech	$0.03_{\pm0.002}(44)$	$0.054_{\pm0.007}(6)$	$0.042_{\pm0.007}(15)$	$0.037_{\pm0.007}(31)$	$0.043_{\pm0.008}(13)$	$0.056_{\pm0.015}(4)$	$0.038_{\pm0.001}(22)$	$0.031_{\pm0.006}(42)$	$0.036_{\pm0.006}(32)$	$0.036_{\pm0.007}(32)$	$0.031_{\pm0.006}(42)$	$0.038_{\pm0.001}(22)$
Avg Ranking	18.11	6.56	17.56	32.5	6.5	8.56	11.83	33.0	27.14	15.11	33.44	11.0

Dataset	CBLOF	CD	ECOD	FB	GMM	HBOS	IForest	KDE	LMDD	LODA	MCD	OCSVM
backdoor	$0.095_{\pm0.013}(23)$	$0.242_{\pm0.19}(16)$	$0.169_{\pm0.001}(18)$	$0.556_{\pm0.099}(13)$	$0.87_{\pm0.001}(2)$	$0.086_{\pm0.001}(26)$	$0.089_{\pm0.01}(25)$	$0.455_{\pm0.008}(14)$	NAN	$0.072_{\pm0.07}(30)$	$0.202_{\pm0.089}(17)$	$0.077_{\pm0.003}(29)$
campaign	$0.488_{\pm0.006}(16)$	$0.469_{\pm0.025}(20)$	$0.502_{\pm0.002}(4)$	$0.307_{\pm0.067}(39)$	$0.517_{\pm0.003}(1)$	$0.508_{\pm0.005}(2)$	$0.462_{\pm0.02}(22)$	$0.478_{\pm0.003}(19)$	$0.429_{\pm0.029}(27)$	$0.302_{\pm0.074}(40)$	$0.482_{\pm0.014}(18)$	$0.497_{\pm0.003}(8)$
census	$0.202_{\pm0.003}(10)$	$0.259_{\pm0.186}(1)$	NAN	$0.126_{\pm0.002}(32)$	$0.2_{\pm0.002}(11)$	$0.139_{\pm0.0}(28)$	$0.14_{\pm0.004}(27)$	$0.216_{\pm0.006}(4)$	NAN	$0.135_{\pm0.049}(30)$	NAN	$0.205_{\pm0.003}(9)$
InternetAds	$0.529_{\pm0.109}(21)$	$0.315_{\pm0.0}(35)$	$0.619_{\pm0.009}(15)$	$0.565_{\pm0.097}(17)$	$0.775_{\pm0.068}(7)$	$0.26_{\pm0.027}(39)$	$0.265_{\pm0.037}(38)$	$0.773_{\pm0.05}(8)$	$0.399_{\pm0.091}(26)$	$0.397_{\pm0.137}(27)$	NAN	$0.546_{\pm0.115}(18)$
mnist	$0.684_{\pm0.011}(14)$	$0.281_{\pm0.12}(41)$	$0.303_{\pm0.007}(40)$	$0.716_{\pm0.025}(11)$	$0.725_{\pm0.012}(9)$	$0.215_{\pm0.005}(44)$	$0.539_{\pm0.043}(28)$	$0.774_{\pm0.018}(2)$	$0.46_{\pm0.112}(32)$	$0.419_{\pm0.115}(35)$	$0.583_{\pm0.093}(22)$	$0.683_{\pm0.008}(15)$
musk	$1.0_{\pm0.0}(1)$	$0.132_{\pm0.005}(43)$	$0.604_{\pm0.005}(33)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$1.0_{\pm0.0}(1)$	$0.355_{\pm0.052}(37)$	$1.0_{\pm0.0}(1)$	$0.992_{\pm0.007}(24)$	$0.997_{\pm0.003}(22)$	$0.987_{\pm0.017}(26)$	$1.0_{\pm0.0}(1)$
optdigits	$0.14_{\pm0.015}(18)$	$0.042_{\pm0.002}(42)$	$0.066_{\pm0.001}(31)$	$0.511_{\pm0.024}(3)$	$0.132_{\pm0.003}(19)$	$0.384_{\pm0.029}(6)$	$0.161_{\pm0.048}(17)$	$0.489_{\pm0.031}(4)$	$0.052_{\pm0.002}(39)$	$0.057_{\pm0.015}(35)$	$0.072_{\pm0.007}(28)$	$0.068_{\pm0.002}(29)$
SpamBase	$0.846_{\pm0.01}(17)$	$0.537_{\pm0.015}(43)$	$0.682_{\pm0.008}(34)$	$0.732_{\pm0.028}(30)$	$0.837_{\pm0.011}(20)$	$0.781_{\pm0.014}(27)$	$0.854_{\pm0.021}(12)$	$0.883_{\pm0.004}(4)$	$0.704_{\pm0.042}(32)$	$0.769_{\pm0.067}(28)$	$0.814_{\pm0.058}(25)$	$0.847_{\pm0.012}(16)$
speech	$0.038_{\pm0.001}(22)$	$0.038_{\pm0.001}(22)$	$0.04_{\pm0.002}(17)$	$0.044_{\pm0.001}(11)$	$0.056_{\pm0.007}(4)$	$0.044_{\pm0.006}(11)$	$0.032_{\pm0.002}(41)$	$0.065_{\pm0.006}(1)$	$0.045_{\pm0.008}(10)$	$0.033_{\pm0.005}(38)$	$0.043_{\pm0.002}(13)$	$0.038_{\pm0.001}(22)$
Avg Ranking	15.78	29.22	24.0	17.44	8.22	20.44	27.44	6.33	27.14	31.67	21.29	16.33

Dataset	QMCD	Sampling	ABOD	COF	INNE	KNN	KPCA	LOF	PCA
backdoor	$0.051_{\pm0.0}(36)$	$0.081_{\pm0.012}(28)$	$0.095_{\pm0.001}(23)$	NAN	$0.072_{\pm0.006}(30)$	$0.109_{\pm0.001}(21)$	NAN	$0.107_{\pm0.001}(22)$	$0.048_{\pm0.0}(38)$
campaign	$0.496_{\pm0.012}(10)$	$0.466_{\pm0.036}(21)$	$0.347_{\pm0.003}(36)$	NAN	$0.392_{\pm0.005}(32)$	$0.413_{\pm0.003}(28)$	NAN	$0.313_{\pm0.001}(38)$	$0.202_{\pm0.0}(43)$
census	$0.207_{\pm0.001}(7)$	$0.199_{\pm0.003}(13)$	$0.094_{\pm0.0}(37)$	NAN	$0.172_{\pm0.003}(21)$	$0.181_{\pm0.0}(17)$	NAN	$0.137_{\pm0.001}(29)$	$0.117_{\pm0.0}(34)$
InternetAds	$0.315_{\pm0.0}(35)$	$0.536_{\pm0.105}(19)$	$0.339_{\pm0.011}(32)$	$0.227_{\pm0.01}(41)$	$0.325_{\pm0.025}(34)$	$0.366_{\pm0.032}(28)$	$0.328_{\pm0.019}(33)$	$0.207_{\pm0.006}(42)$	$0.306_{\pm0.004}(37)$
mnist	$0.358_{\pm0.01}(37)$	$0.679_{\pm0.021}(16)$	$0.445_{\pm0.009}(33)$	$0.317_{\pm0.016}(39)$	$0.543_{\pm0.02}(27)$	$0.555_{\pm0.014}(24)$	$0.269_{\pm0.018}(42)$	$0.347_{\pm0.013}(38)$	$0.169_{\pm0.0}(46)$
musk	$0.032_{\pm0.0}(46)$	$1.0_{\pm0.0}(1)$	$0.042_{\pm0.006}(45)$	$0.141_{\pm0.015}(41)$	$0.931_{\pm0.031}(28)$	$0.191_{\pm0.003}(40)$	$0.279_{\pm0.143}(39)$	$0.326_{\pm0.009}(38)$	$0.98_{\pm0.007}(27)$
optdigits	$0.031_{\pm0.0}(46)$	$0.127_{\pm0.006}(20)$	$0.098_{\pm0.002}(22)$	$0.061_{\pm0.001}(33)$	$0.051_{\pm0.003}(40)$	$0.036_{\pm0.0}(43)$	$0.054_{\pm0.011}(37)$	$0.068_{\pm0.002}(29)$	$0.056_{\pm0.0}(36)$
SpamBase	$0.751_{\pm0.03}(29)$	$0.843_{\pm0.011}(19)$	$0.604_{\pm0.007}(38)$	$0.499_{\pm0.008}(45)$	$0.674_{\pm0.016}(35)$	$0.725_{\pm0.01}(31)$	$0.648_{\pm0.011}(36)$	$0.485_{\pm0.009}(46)$	$0.522_{\pm0.004}(44)$
speech	$0.026_{\pm0.0}(46)$	$0.035_{\pm0.001}(34)$	$0.049_{\pm0.01}(7)$	$0.029_{\pm0.003}(45)$	$0.035_{\pm0.002}(34)$	$0.038_{\pm0.001}(22)$	$0.033_{\pm0.004}(38)$	$0.04_{\pm0.002}(17)$	$0.038_{\pm0.001}(22)$
Avg Ranking	32.44	19.0	30.33	40.67	31.22	28.22	37.5	33.22	36.33